

Skolkovo Institute of Science and Technology

As a manuscript

Alexander Ivanovich Panchenko

**Methods and Algorithms of Computational Lexical Semantics
with Applications to Extraction of Senses, Hyponyms, and Frames**

Dissertation Summary

for the purpose of obtaining academic degree

Doctor of Science in Computer Science

Moscow – 2024

This doctoral dissertation was prepared at Skolkovo Institute of Science and Technology (Skoltech) and partially at Artificial Intelligence Research Institute (AIRI) based on publications prepared at University of Hamburg (UHH), Technical University of Darmstadt (TUDA), Skoltech, and AIRI. The earliest publication related to this dissertation dates back to 2016 while the latest one has been published in 2023. The current document provides a summary of scientific contributions as well as full texts of the original papers and articles in presented in separate appendices.

Contents

1	Introduction	5
2	Graph Clustering for Sense and Frame Induction	22
2.1	Introduction	22
2.2	Method	22
2.3	Results	30
3	Word Sense Embeddings	32
3.1	Introduction	32
3.2	Method	33
3.3	Results	39
4	Unsupervised Interpretable Word Sense Disambiguation	41
4.1	Introduction	41
4.2	Method	43
4.3	Results	45
5	Linking Word Sense Representations	47
5.1	Introduction	47
5.2	Method	48
5.3	Results	52
6	Prediction of Hypernym Embeddings	53
6.1	Introduction	53

6.2	Method	53
6.3	Results	55
7	Extracting of Hypernyms via Sense Graph Clustering	56
7.1	Introduction	56
7.2	Method	57
7.3	Results	60
8	Taxonomy Enrichment using Hyperbolic Embeddings	61
8.1	Introduction	61
8.2	Method	62
8.3	Results	64
9	Node Embeddings of Lexical-Semantic Graphs	66
9.1	Introduction	66
9.2	Method	67
9.3	Results	69
10	Lexical Substitution and Analysis of its Semantic Relation Types	70
10.1	Introduction	70
10.2	Method	71
10.3	Results	73
11	Conclusion	74
	Bibliography	78
A	Copies of published research papers and journal articles	90
A.1	Paper “Unsupervised Does Not Mean Uninterpretable: The Case for Word Sense Induction and Disambiguation”	90
A.2	Paper “Always Keep your Target in Mind: Studying Semantics and Improving Performance of Neural Lexical Substitution”	104

A.3	Paper “Making Fast Graph-based Algorithms with Graph Metric Embeddings”	119
A.4	Paper “Negative Sampling Improves Hypernymy Extraction Based on Projection Learning”	127
A.5	Paper “Every child should have parents: a taxonomy refinement algorithm based on hyperbolic term embeddings”	136
A.6	Article “Watset: Local-Global Graph Clustering with Applications in Sense and Frame Induction”	144
A.7	Paper “Linked Disambiguated Distributional Semantic Networks” . .	203
A.8	Article “A framework for enriching lexical semantic resources with distributional semantics”	213
A.9	Paper “Watset: Automatic Induction of Synsets from a Graph of Synonyms”	262
A.10	Paper “Unsupervised Semantic Frame Induction using Triclustering”	275
A.11	Paper “Making Sense of Word Embeddings”	284
A.12	Paper “Unsupervised Does Not Mean Uninterpretable: The Case for Word Sense Induction and Disambiguation”	295
A.13	Paper “Improving Hypernymy Extraction with Distributional Semantic Classes”	309
A.14	Paper “Word Sense Disambiguation for 158 Languages using Word Embeddings Only”	321

Chapter 1

Introduction

The traditional lexical resources, such as Wordnets¹, taxonomies, thesauri or dictionaries contain *precise* manually-encoded information about lexical items (such as words and phrases) and relations between them (such as synonyms and hypernyms) yet coverage i.e. *recall* and actuality of these resources is often inherently limited. This is due to the expensive, long and usually purely manual process of resource creation and keeping it up-to-date. Besides, some domain-specific terms may be simply out of scope even for the largest lexicographical collaboratively created resources, such as Wiktionary.²

On the other hand, it is possible to apply data-driven approaches, such as distributional semantic models and information extraction, to mine for word senses and relations between them from large textual corpora, such as Wikipedia³ or CommonCrawl⁴. This alternative way of building lexical semantic resources, in contrast to the manual approach, usually yield high *recall* due to the huge lexical coverage of unlabeled text corpora, but its results may be noisy i.e. of low *precision*. One of the overreaching goals of this work was to bridge the gap between these two views on lexical semantics getting the best of both worlds while combining high precision of manual resources with high recall of automatic techniques.

¹<https://wordnet.princeton.edu>

²<https://www.wiktionary.org>

³<https://www.wikipedia.org>

⁴<https://commoncrawl.org>

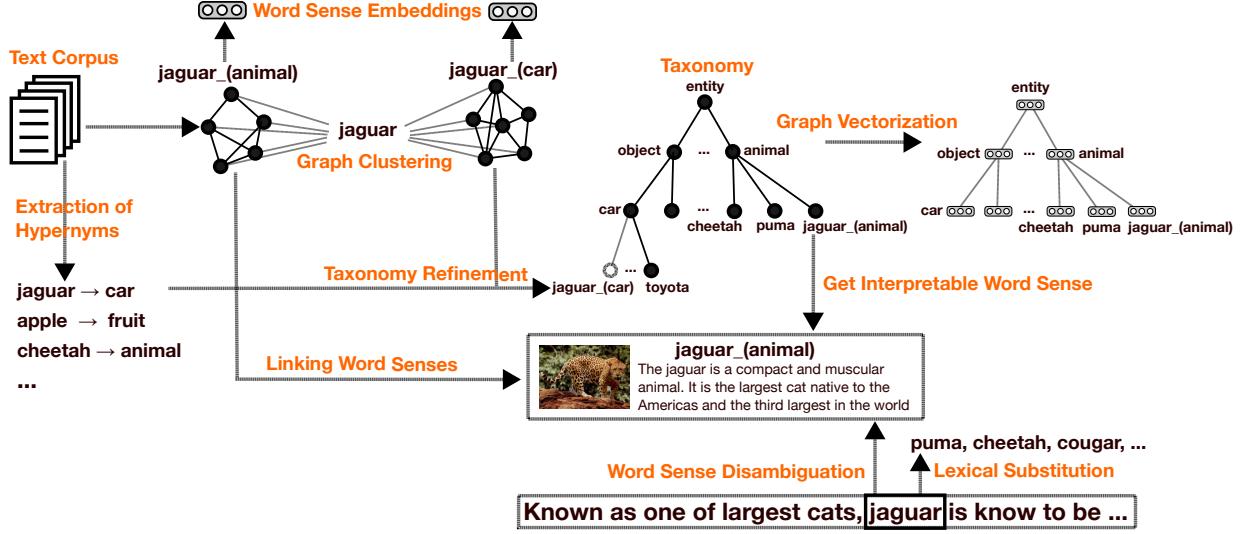


Figure 1-1: Overview of various methods for computational lexical semantics presented in this dissertation and their interrelations.

Figure 1-1 presents an overview of various methods for computational lexical semantics proposed in this dissertation and illustrate how they interact with one another. The *linking of word senses* action highlighted in Figure 1-1 is the method supposed to bridge this gap. Namely, word senses induced from text corpus though *graph clustering* is the process known as *word sense induction*. Word senses mined from text though this procedure may already be present in the lexical resource, such as “jaguar_(animal)”. In this case, a linking of these senses is required. Alternatively a word sense may cover a new meanings not present in the resource, but represented in text. In the latter case, this newly discovered word sense is added into the resource increasing its recall. Similar is done with semantic relations. The figure illustrates how hypernyms (also known as “Is-A” or hierarchical relations) are extracted from text corpus and used to perform enrichment of an existing taxonomy.

Now let us turn to another gap between two mentioned above methodological “realms” (i.e., manually created vs data-driven lexical representations of words), namely the lack of explicit sense annotations in raw texts. At the very bottom of Figure 1-1 a textual context of the word “jaguar” is presented. A string “jaguar” may refer to several word senses, such as “jaguar_(animal)” and “jaguar_(car)” as described in a manually created taxonomy. However, there is obviously no identifier

in text referring to one of these senses in the lexical resource. A *word sense disambiguation* (WSD) algorithm automatically identifies the most suitable meaning of a word given the context. Another related technology is *lexical substitution*: instead of explicit linking to word senses it generates semantically suitable substitutes (usually, synonyms, co-hyponyms, or hypernyms) in the correct sense. For instance, here for the word “jaguar” substitutes “puma”, “cheetah”, and “cougar” are generated and not “Mercedes”, “BMW”, or “Audi”.

The interest of performing WSD is to link text to rich, precise and interpretable word sense representations from the manually created resource, which may feature elements such as definitions, images, hypernyms, and related words. At the same time, newly inserted word senses discovered automatically as described above, do not have such representations. That is why one of the contributions was to create a technology for automatic building such rich human-interpretable multi-modal representations.

Finally, the use of word sense representations in classic machine learning models, but especially in (deep) neural networks is complicated if these are represented in symbolic form of graphs (which correspond to one-hot encoding). Besides, for similarity computations between word senses low-dimensional dense vector format (also known as “embedding”) is preferable over sparse vector format or graph representations. That is why, procedures of *graph vectorization* were developed and tested on large scale lexical resources, such as Wordnet and DBpedia⁵. Taking as input a lexical resource network these output node embeddings of this graph. Also procedures for obtaining *word sense embeddings* from the graph-based sense representations induced from text were proposed. These representations are further used to perform WSD based on similarity computations between context words and these learned vector representations of word senses.

Contributions presented in this dissertation cover a wide range of task related to lexical computational semantics: they form a solid methodological framework for learning, population, linking, disambiguation and vectorization of word senses and relations between them.

⁵<https://dbpedia.org>

Methodologically, many developed methods are graph-based, more specifically graph clustering algorithms, including newly proposed, are used to process linguistic networks of various kinds. The use of graph representation is fairly natural as each lexical semantic resource can be represented as graph with nodes being word senses / terms and edges being semantic relations between them. As the modern NLP methods heavily rely on neural networks, dealing with such linguistic graphs required their vectorisation. Towards this end, methods for node embedding of linguistic graphs, such as WordNets and Knowledge Graphs (KG) were developed to solve various tasks, such as completion of linguistic resources and word sense disambiguation.

The proposed methods for learning and population from text are applicable to all common lexical linguistic resources, such as lexical semantic databases, e.g. Wordnet or Babelnet⁶, thesauri, taxonomies, as they all can be represented in the form of graph with nodes corresponding to word senses. Methods for induction of word senses from text mining new nodes of such graphs. Algorithms for linking of newly mined word senses with existing resources find correspondances between nodes of manually constructed and automatically constructed graphs. Methods for induction of semantic taxonomies, which form a backbone of pretty much every type of lexical semantic resource, are learning special kinds of directed graphs - trees. Crucially, most practical application of a sophisticated lexical resource, such as WordNet, involve mapping senses representation listed in this resource to tokens occurring in a raw text. Towards this end, developed methods for WSD search for optimal correspondences in terms of semantic coherence of nodes of lexical resource and tokens in text.

Therefore, the set of proposed methods deal with both graph and vector-based representation of word senses as they are highly complementary: for various use-cases one or another may be used. The contents of the thesis are arranged in 9 “content chapters”, covering a variety of topics. At the same time each chapter is related to the use of graph and/or vector based representations to a lexical-semantic processing task. Among all methodological techniques graph clustering shall be highlighted as the central one, applied in various contexts, such as induction of word

⁶<https://babelnet.org>

senses, semantic frames or improving quality of hypernymy relations.

Object and goals of the dissertation.

The purpose of the dissertation is the development of methods for computational lexical semantics which would bridge the gap between (i) the *precise well-interpretable* manually created lexical resources with *low lexical coverage* and (ii) *noisy non-interpretable* automatically induced from text distributional lexical representations with *high lexical coverage*. This includes (i) development of new algorithms for processing large linguistic networks constructed from both manually created lexical resources and graphs induced from text, (ii) development of method for induction of lexical semantic structures of various kinds from text, most notably word senses, (iii) development of techniques for making the induced structures interpretable in the way they are in manually constructed resources, (iv) development of methods for effective disambiguation in context with respect to the induced sense representations, (v) development of effective vectorization of lexical semantic graphs for the use in various application.

The obtained results:

1. Developing an algorithm for fuzzy graph clustering effective and efficient for processing of large linguistic network data (Chapter 2).
2. Based on the developed fuzzy graph clustering algorithm, we propose a methods for induction of three lexical semantic structures: synsets⁷, semantic frames, and semantic classes (Chapter 2).
3. Proposing a method for learning sense embeddings from word embeddings using graph clustering, and a word sense disambiguation method using the induced sense representations (Chapter 3).
4. Developing a method for inducing from text word sense representations using graph-based distributional methods and techniques for making these

⁷Synset is a set of synonyms, e.g. {car, vehicle, automobile}. These can be represented as a clique in terms of graph representation with synonymy relationships as edges.

representations interpretable by automatic retrieval of hypernyms, images, and definitions (Chapter 4).

5. Proposing a framework for enriching lexical resources with distributional information, featuring algorithm for linking distributionally induced sense representations to manually created lexical resources, and algorithm for disambiguation of related words and hypernyms (Chapter 5).
6. Developing a model for generation of hypernyms of words based on projection learning with regularization of relation asymmetry (Chapter 6).
7. Proposing a method for post-processing of hypernymy relations using distributionally induced semantic classes: wrong hypernyms are removed while the missing ones are added (Chapter 7).
8. Proposing an algorithm for construction of taxonomic tree (composed of binary hypernymy relations between terms) using Euclidean and hyperbolic (Poincaré) vector representations (Chapter 8).
9. Proposing a model for learning node embeddings of linguistic networks using supervision from graph-based similarity metrics and show its applicability to word sense disambiguation task, *inter alia* (Chapter 9).
10. Proposing methods for neural lexical substitution which integrates information about the target word with the information about the context (Chapter 10).
11. Investigating distribution of lexical semantic relations provided the neural lexical semantic models, such as synonyms, hypernyms, co-hyponyms⁸, etc (Chapter 10).

Author's contribution includes the formal problem formulations and experimental design for all the mentioned above results, the design of the methods and algorithms mentioned, analysis and generalization of the results. The more

⁸Co-hyponyms are terms with common hypernyms, e.g. “apple → fruit” and “pear → fruit”.

specific author's contribution (and list of key collaborators) related to each paper are provided in the list of publications below.

The novelty of the proposed research lies in the development of new algorithms for computational lexical semantics. Most of the proposed methods deal with either distributional semantics methods, graph clustering algorithms or both of them. A novel graph clustering approach lies at the core of several newly proposed methods for lexical semantics. For instance, those for automatic induction of lexical semantic structures, such as synsets, semantic frames and semantic classes. Several contributions are related to automatic processing of hypernymy relationships between words. In particular, in the dissertation the author proposes:

- Algorithm for fuzzy graph clustering (Chapter 2);
- Methods for induction of synsets, semantic classes and semantic frames (Chapter 2);
- Methods for learning word sense embeddings (Chapter 3);
- Method for inducing interpretable word sense representations from text (Chapter 4);
- Method for linking distributional word sense representations with lexical resources (Chapter 5);
- Model for generation of hypernyms (Chapter 6);
- Method for post-processing of noisy hypernymy relations (Chapter 7);
- Algorithm for construction of lexical semantic trees i.e. taxonomies (Chapter 8);
- Model for node embeddings of linguistic graphs (Chapter 9);
- Methods for neural lexical substitution (Chapter 10);
- Study of distribution of lexical semantic relations provided by neural lexical substitution models (Chapter 10).

The scope of dissertation is covered in 42 publications [1–42], among those:

- 5 papers are published in CORE A* conferences [3, 5, 9, 10, 13];
- 6 papers are published in CORE A conferences [1, 2, 4, 7, 16];
- 5 articles are published in Q1 journals [6, 8, 11, 12, 15];
- 1 paper is published in CORE A* conference student track [28];
- 1 paper is published in CORE A conference demo track [18];
- 5 papers are published at CORE B conference [19, 20, 26, 27, 29];
- 11 papers indexed by Scopus published in proceedings of the main volumes of conferences [22–25, 30–34, 40, 41];
- 8 papers indexed by Scopus published in workshops co-located with top conferences (CORE A*/A) [17, 21, 35–39, 42].

According to regulations of the Dissertation Council in Computer Sciences of Higher School of Economics and for brevity among the 42 papers topically fitting the dissertation only 20 key papers are listed below including *all* 16 first-tier, i.e. A*/A/Q1, and *selected* 4 second-tier. The *remaining* 22 second-tier publications are cited in the bibliography and are openly available online. The defence is performed based on 14 publications of these 20 listed below works. Namely, based on the first 10 from the list of first-tier publications [1–10] and the 4 second-tier publications [17–20]. To avoid confusions, an note “used for defence” is included below where appropriate.

First-tier publications:

1. A. Panchenko, E. Ruppert, S. Faralli, S. P. Ponzetto, and C. Biemann, “**Unsupervised does not mean uninterpretable: The case for word sense induction and disambiguation**,” in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, (Valencia, Spain), pp. 86–98, Association for Computational Linguistics, Apr. 2017

<https://aclanthology.org/E17-1009>

[CORE A] used for defence; main co-author; the author of this thesis has proposed an interpretable unsupervised knowledge-free word sense disambiguation (WSD) method consisting of (1) a technique to disambiguation that relies on induced inventories as a pivot for learning sense feature representations, (2) a technique for making induced sense representations interpretable by labelling them with hypernyms and images.

2. N. Arefyev, B. Sheludko, A. Podolskiy, and A. Panchenko, “**Always Keep your Target in Mind: Studying Semantics and Improving Performance of Neural Lexical Substitution**,” in *Proceedings of the 28th International Conference on Computational Linguistics*, (Barcelona, Spain (Online)), pp. 1242–1255, International Committee on Computational Linguistics, Dec. 2020

<https://aclanthology.org/2020.coling-main.107>

[CORE A] used for defence; main co-author; the author designed (in an inseparable cooperation with N. Arefyev) methods of target word injection for lexical substitution quality improvement; performed an analysis of types of semantic relations (synonyms, hypernyms, co-hyponyms, etc.) produced by neural substitution models.

3. A. Kutuzov, M. Dorgham, O. Oliynyk, C. Biemann, and A. Panchenko, “**Making Fast Graph-based Algorithms with Graph Metric Embeddings**,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 3349–3355, Association for Computational Linguistics, July 2019

<https://aclanthology.org/P19-1325>

[CORE A*] used for defence; main co-author; the author designed a method that learns dense vector embeddings of nodes based on a pre-defined graph-based similarity measure, e.g. the shortest path distance.

4. D. Ustalov, N. Arefyev, C. Biemann, and A. Panchenko, “**Negative Sampling Improves Hypernymy Extraction Based on Projection Learning**,” in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, (Valencia, Spain), pp. 543–550,

Association for Computational Linguistics, Apr. 2017

<https://aclanthology.org/E17-2087>

[CORE A] used for defence; main co-author; the author designed an approach for hypernymy extraction based on projection learning, which makes use of both positive and negative training instances enforcing the asymmetry of the projection.

5. R. Aly, S. Acharya, A. Ossa, A. Köhn, C. Biemann, and A. Panchenko, “**Every Child Should Have Parents: A Taxonomy Refinement Algorithm Based on Hyperbolic Term Embeddings**,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 4811–4817, Association for Computational Linguistics, July 2019

<https://aclanthology.org/P19-1474>

[CORE A*] used for defence; main co-author; the author designed a method for refinement of semantic tree structures (taxonomies) using embeddings based on Euclidian and hyperbolic vector spaces.

6. D. Ustalov, A. Panchenko, C. Biemann, and S. P. Ponzetto, “**Watset: Local-Global Graph Clustering with Applications in Sense and Frame Induction**,” *Computational Linguistics*, vol. 45, pp. 423–479, Sept. 2019

<https://aclanthology.org/J19-3002>

[Q1] used for defence; main co-author; the author proposed (in an inseparable cooperation with D. Ustalov) a meta-algorithm for fuzzy graph clustering using hard clustering methods; methods for induction of synsets, semantic frames, and semantic classes based on based on the proposed algorithm.

7. S. Faralli, A. Panchenko, C. Biemann, and S. P. Ponzetto, “**Linked Disambiguated Distributional Semantic Networks**,” in *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part II* (P. Groth, E. Simperl, A. J. G. Gray, M. Sabou, M. Krötzsch, F. Lécué, F. Flöck, and Y. Gil, eds.), vol. 9982 of *Lecture Notes in Computer Science*, pp. 56–64, 2016

https://doi.org/10.1007/978-3-319-46547-0__7

[CORE A] used for defence; main co-author; the author developed (in inseparable cooperation with S. Faralli) methodology to automatically induce

distributionally-based semantic representations from large amounts of text, and link them to a reference knowledge base.

8. C. Biemann, S. Faralli, A. Panchenko, and S. P. Ponzetto, “**A framework for enriching lexical semantic resources with distributional semantics**,” *Nat. Lang. Eng.*, vol. 24, no. 2, pp. 265–312, 2018
<https://doi.org/10.1017/S135132491700047X>
[Q1] *used for defence; main co-author; the author has proposed (in inseparable cooperation with S. Faralli) a framework for enriching lexical semantic resources consisting of methodos for combining information from distributional semantic models with manually constructed lexical semantic resources.*
9. D. Ustalov, A. Panchenko, and C. Biemann, “**Watset: Automatic Induction of Synsets from a Graph of Synonyms**,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Vancouver, Canada), pp. 1579–1590, Association for Computational Linguistics, July 2017
<https://aclanthology.org/P17-1145>
[CORE A*] *used for defence; main co-author; the author has proposed (in inseparable cooperation with D. Ustalov) a novel approach that resolves ambiguities in the input graph to perform fuzzy clustering.*
10. D. Ustalov, A. Panchenko, A. Kutuzov, C. Biemann, and S. P. Ponzetto, “**Unsupervised Semantic Frame Induction using Triclustering**,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, (Melbourne, Australia), pp. 55–62, Association for Computational Linguistics, July 2018
<https://aclanthology.org/P18-2010>
[CORE A*] *used for defence; main co-author; the author has proposed (in inseparable cooperation with D. Ustalov) new approach to triclustering; proposed to apply triclustering algorithms for unsupervised frame induction; proposed new method for the evaluation of frame induction.*
11. Ö. Sevgili, A. Shelmanov, M. Y. Arkhipov, A. Panchenko, and C. Biemann, “**Neural**

- entity linking: A survey of models based on deep learning,” *Semantic Web*, vol. 13, no. 3, pp. 527–570, 2022
<https://doi.org/10.3233/SW-222986>
- [Q1] main co-author; the author has proposed the overall idea and structure of the survey; created formal definitions of tasks; guided the writing.
12. S. Anwar, A. Shelmanov, N. Arefyev, [A. Panchenko](#), and C. Biemann, “Text augmentation for semantic frame induction and parsing,” *Language Resources and Evaluation*, vol. 23, no. 3, pp. 527–556, 2023
<https://link.springer.com/article/10.1007/s10579-023-09679-8>
- [Q1] main co-author; the author designed a one-shot method for inducing frame-semantic structures using lexical substitution on frame-annotated sentences; designed experimental setting for semantic frame parsing and induction.
13. A. Jana, D. Puzyrev, [A. Panchenko](#), P. Goyal, C. Biemann, and A. Mukherjee, “On the Compositionality Prediction of Noun Phrases using Poincaré Embeddings,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 3263–3274, Association for Computational Linguistics, July 2019
<https://aclanthology.org/P19-1316>
- [CORE A*] main co-author; the author designed a straightforward and efficient approach for combining distributional and hypernymy information using hyperbolic embeddings for the task of noun phrase compositionality prediction.
14. I. Nikishina, V. Logacheva, [A. Panchenko](#), and N. Loukachevitch, “Studying Taxonomy Enrichment on Diachronic WordNet Versions,” in *Proceedings of the 28th International Conference on Computational Linguistics*, (Barcelona, Spain (Online)), pp. 3095–3106, International Committee on Computational Linguistics, Dec. 2020
<https://aclanthology.org/2020.coling-main.276>
- [CORE A] main co-author; the author designed several methods for taxonomy enrichment.
15. I. Nikishina, M. Tikhomirov, V. Logacheva, Y. Nazarov, [A. Panchenko](#), and N. V.

Loukachevitch, “**Taxonomy enrichment with text and graph vector representations**,” *Semantic Web*, vol. 13, no. 3, pp. 441–475, 2022

<https://doi.org/10.3233/SW-212955>

[Q1] *main co-author; the author designed (in inseparable cooperation with I. Nikishina) methods for taxonomy enrichment using text and graph vectors.*

16. S. Faralli, A. Panchenko, C. Biemann, and S. P. Ponzetto, “**The ContrastMedium Algorithm: Taxonomy Induction From Noisy Knowledge Graphs With Just A Few Links**,” in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, (Valencia, Spain), pp. 590–600, Association for Computational Linguistics, Apr. 2017
<https://aclanthology.org/E17-1056>
[CORE A] *main co-author; the author designed (in inseparable cooperation with S. Faralli) end-to-end pipeline for taxonomy induction from scratch using distributional semantic representations.*

Second-tier publications:

17. M. Pelevina, N. Arefiev, C. Biemann, and A. Panchenko, “**Making Sense of Word Embeddings**,” in *Proceedings of the 1st Workshop on Representation Learning for NLP*, (Berlin, Germany), pp. 174–183, Association for Computational Linguistics, Aug. 2016
<https://aclanthology.org/W16-1620>
[Scopus, highly cited] used for defence; *main co-author; the author of this thesis designed a novel method for learning sense embeddings from word embeddings and a word sense disambiguation (WSD) mechanism using these sense representations.*
18. A. Panchenko, F. Marten, E. Ruppert, S. Faralli, D. Ustalov, S. P. Ponzetto, and C. Biemann, “**Unsupervised, Knowledge-Free, and Interpretable Word Sense Disambiguation**,” in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, (Copenhagen, Denmark), pp. 91–96, Association for Computational Linguistics, Sept. 2017
<https://aclanthology.org/E17-1009>

[CORE A, demo track] *used for defence; main co-author; the author of this thesis has designed the first system for word sense induction and disambiguation, which is unsupervised, knowledge-free, and interpretable at the same time.*

19. A. Panchenko, D. Ustalov, S. Faralli, S. P. Ponzetto, and C. Biemann, “**Improving Hypernymy Extraction with Distributional Semantic Classes**,” in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, (Miyazaki, Japan), European Language Resources Association (ELRA), May 2018
<https://aclanthology.org/L18-1244>

[CORE B] *used for defence; main co-author; the author of this thesis has proposed an unsupervised method for post-processing of noisy hypernymy relations based on clustering of graphs of word senses induced from text.*

20. V. Logacheva, D. Teslenko, A. Shelmanov, S. Remus, D. Ustalov, A. Kutuzov, E. Artemova, C. Biemann, S. P. Ponzetto, and A. Panchenko, “**Word Sense Disambiguation for 158 Languages using Word Embeddings Only**,” in *Proceedings of the Twelfth Language Resources and Evaluation Conference*, (Marseille, France), pp. 5943–5952, European Language Resources Association, May 2020
<https://aclanthology.org/2020.lrec-1.728>

[CORE B] *used for defence; main co-author; the author of this thesis proposed a new algorithm of unsupervised word sense induction, which creates sense inventories based on pre-trained word vectors which is based on ego-graph vector induction.*

Reports at conferences and seminars:

In this section, we list conferences where the 42 papers relevant to the scope of this dissertation were presented, not only those 14 papers the defence is based on.⁹

1. **ACL-2019** [CORE A*] [3,5,13,28,36]: The 57th Annual Meeting of the Association for Computational Linguistics, (Florence, Italy), Association for Computational Linguistics, July 2019.

⁹To obtain the exact rank of a publication please refer to the list above as some of the papers were presented at the associated workshops co-located with the main conference.

2. **ACL-2018** [CORE A*] [10]: The 56th Annual Meeting of the Association for Computational Linguistics (Melbourne, Australia), Association for Computational Linguistics, July 2018.
3. **ACL-2017** [CORE A*] [9]: The 55th Annual Meeting of the Association for Computational Linguistics (Vancouver, Canada), Association for Computational Linguistics, July 2017.
4. **ACL-2016** [CORE A*] [17]: The 54th Annual Meeting of the Association for Computational Linguistics (Berlin, Germany). Association for Computational Linguistics.
5. **IJCNLP-ACL-2021** [CORE A*] [35]: The Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Bangkok, Thailand), Association for Computational Linguistics.
6. **COLING-2022** [CORE A] [42]: The 29th International Conference on Computational Linguistics (Gyeongju, Republic of Korea), International Committee on Computational Linguistics, October 2022.
7. **COLING-2020** [CORE A] [2, 14]: The 28th International Conference on Computational Linguistics, (Barcelona, Spain), International Committee on Computational Linguistics, December 2020.
8. **EACL-2017** [CORE A] [1, 4, 16, 39]: The 15th Conference of the European Chapter of the Association for Computational Linguistics (Valencia, Spain), Association for Computational Linguistics, April 2017. [1]
9. **EMNLP-2017** [CORE A] [18]: The 2017 Conference on Empirical Methods in Natural Language Processing (Copenhagen, Denmark), Association for Computational Linguistics, September 2017.
10. **ISWC-2016** [CORE A] [7]: The 15th International Semantic Web Conference, (Kobe, Japan), vol. 9982 of Lecture Notes in Computer Science, October, 2016.

11. **NAACL-2019** [CORE A] [37, 38]: 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Minneapolis, Minnesota, USA), Association for Computational Linguistics, June 2019.
12. **NAACL-2016** [CORE A] [21]: The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (San Diego, California, USA), Association for Computational Linguistics.
13. **AACL-2022** [CORE B] [40]: The 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Taipei, Taiwan), Association for Computational Linguistics, November 2022.
14. **LREC-2020** [CORE B] [20]: The 12th Language Resources and Evaluation Conference, (Marseille, France), European Language Resources Association, May 2020.
15. **LREC-2018** [CORE B] [19, 26, 27]: The 11th International Conference on Language Resources and Evaluation (LREC 2018), (Miyazaki, Japan), European Language Resources Association (ELRA), May 2018.
16. **LREC-2016** [CORE B] [29]: The 10th International Conference on Language Resources and Evaluation (LREC'16), (Portorož, Slovenia), pp. 2649–2655, European Language Resources Association (ELRA), May 2016.
17. **PaM-2020** [Scopus] [22]: The Probability and Meaning Conference (Gothenburg, Sweden), Association for Computational Linguistics, June 2020.
18. **RANLP-2019** [Scopus] [33]: The International Conference on Recent Advances in Natural Language Processing (Varna, Bulgaria), INCOMA Ltd., September 2019.
19. **GWC-2021** [Scopus] [41]: The 11th Global Wordnet Conference (Potchefstroom, South Africa), Global Wordnet Association, January 2021.
20. **AIST-2019** [Scopus/Q2] [32]: The 8th International Conference on Analysis of Images, Social Networks and Texts (Kazan, Russia), July 2019, vol. 11832 of Lecture Notes in Computer Science, Springer.

21. **AIST-2017** [Scopus/Q2] [30]: The 6th International Conference on Analysis of Images, Social Networks and Texts (Moscow, Russia), July 2017, vol. 10716 of Lecture Notes in Computer Science, Springer.
22. **Dialogue-2018** [Scopus] [24, 25]: The 24th International Conference on Computational Linguistics and Intellectual Technologies (Moscow, Russia), RGGU, June 2018.
23. **KONVENS-2018** [Scopus] [23]: The 14th Conference on Natural Language Processing (Vienna, Austria), September, 2018 Österreichische Akademie der Wissenschaften.
24. **KONVENS-2016** [Scopus] [31]: The 13th Conference on Natural Language Processing, (Bochum, Germany), September 2016, vol. 16 of Bochumer Linguistische Arbeitsberichte.

Chapter 2

Graph Clustering for Sense and Frame Induction

Materials of this chapter are based on papers [6, 9, 10] from the list of 14 publications the thesis is based on. Full texts are available in Appendices A.6, A.9, and A.10.

2.1 Introduction

In this chapter, a meta-algorithm for fuzzy graph clustering is presented. It creates an intermediate representation of the input graph that reflects the “ambiguity” of its nodes. Then, it uses hard clustering to discover clusters in this “disambiguated” intermediate graph.

2.2 Method

In this section, we present Watset, a meta-algorithm for fuzzy graph clustering. Given a graph connecting potentially ambiguous objects, e.g., words, it induces a set of unambiguous overlapping clusters (communities) by disambiguating and grouping the ambiguous objects. The meta-algorithm that uses existing *hard* clustering algorithms for graphs to obtain a *fuzzy* clustering, a.k.a. *soft* clustering.

Outline of Fuzzy Graph Clustering Method

Let $G = (V, E)$ be an undirected simple graph, where V is a set of nodes and $E \subseteq V^2$

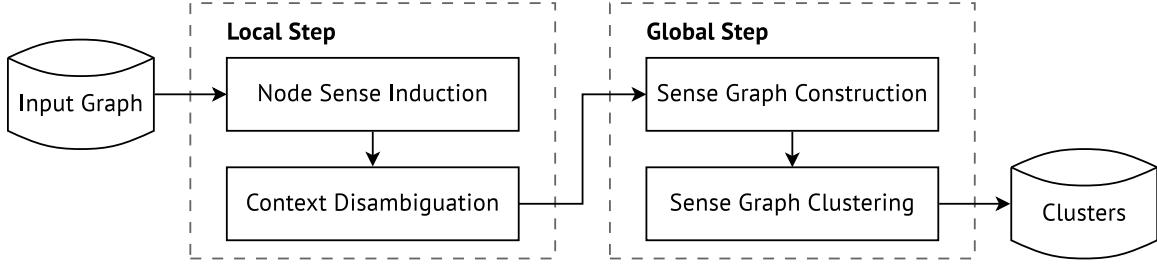


Figure 2-1: The outline of the algorithm showing the *local* step of node sense induction and context disambiguation, and the *global* step of sense graph constructing and clustering.

is a set of undirected edges. We denote a subset of nodes $C^i \subseteq V$ as a cluster. A graph clustering algorithm then is a function $\text{CLUSTER} : (V, E) \rightarrow C$ such that $V = \bigcup_{C^i \in C} C^i$. We distinguish two classes of graph clustering algorithms: *hard* clustering algorithms (partitionings) produce non-overlapping clusters, i.e., $C^i \cap C^j = \emptyset \iff i \neq j$, $\forall C^i, C^j \in C$, while *fuzzy* clustering algorithms permit cluster overlapping, i.e., a node can be a member of several clusters in C .

Algorithm constructs an intermediate representation of the input graph called a *sense graph*. This is achieved by node sense induction based on hard clustering of the input graph node neighborhoods. The sense graph has the edges established between the different *senses* of the input graph nodes. The global clusters of the input graph are obtained by applying a hard clustering algorithm to the sense graph.

An outline of our algorithm is depicted in Figure 2-1: it takes an undirected graph $G = (V, E)$ as the input and outputs a set of clusters C . The algorithm has two steps: local and global. The *local* step, disambiguates the potentially ambiguous nodes in G . The *global* step, uses these disambiguated nodes to construct an intermediate sense graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and produce the overlapping clustering C . Watset is parameterized by two graph partitioning algorithms $\text{Cluster}_{\text{Local}}$ and $\text{Cluster}_{\text{Global}}$, and a context similarity measure sim . The complete pseudocode of Watset is presented in Algorithm 1. For the sake of illustration, while describing the approach, we will provide examples with words and their synonyms. However, Watset is not bound only to the lexical units and relationships, so our examples are given *without loss of generality*. Note also that Watset can be applied for both unweighted and weighted graphs as soon as the underlying hard clustering algorithms $\text{Cluster}_{\text{Local}}$ and $\text{Cluster}_{\text{Global}}$ take edge weights into account.

Algorithm 1 Watset, a Local-Global Meta-Algorithm for Fuzzy Graph Clustering.

Input: graph $G = (V, E)$,
 hard clustering algorithms $\text{Cluster}_{\text{Local}}$ and $\text{Cluster}_{\text{Global}}$,
 context similarity measure $\text{sim} : (\text{ctx}(a), \text{ctx}(b)) \rightarrow \mathbb{R}$, $\forall \text{ctx}(a), \text{ctx}(b) \subseteq V$.

Output: clusters C .

```

1: for all  $u \in V$  do                                 $\triangleright$  Local Step: Sense Induction
2:    $\text{senses}(u) \leftarrow \emptyset$ 
3:    $V_u \leftarrow \{v \in V : \{u, v\} \in E\}$             $\triangleright$  Note that  $u \notin V_u$ 
4:    $E_u \leftarrow \{\{v, w\} \in E : v, w \in V_u\}$ 
5:    $G_u \leftarrow (V_u, E_u)$ 
6:    $C_u \leftarrow \text{Cluster}_{\text{Local}}(G_u)$            $\triangleright$  Cluster the open neighborhood of  $u$ 
7:   for all  $C_u^i \in C_u$  do
8:      $\text{ctx}(u^i) \leftarrow C_u^i$ 
9:      $\text{senses}(u) \leftarrow \text{senses}(u) \cup \{u^i\}$ 
10:   $\mathcal{V} \leftarrow \bigcup_{u \in V} \text{senses}(u)$            $\triangleright$  Global Step: Sense Graph Nodes
11: for all  $\hat{u} \in \mathcal{V}$  do                       $\triangleright$  Local Step: Context Disambiguation
12:    $\widehat{\text{ctx}}(\hat{u}) \leftarrow \emptyset$ 
13:   for all  $v \in \text{ctx}(\hat{u})$  do
14:      $\hat{v} \leftarrow \arg \max_{v' \in \text{senses}(v)} \text{sim}(\text{ctx}(\hat{u}) \cup \{u\}, \text{ctx}(v'))$        $\triangleright \hat{u}$  is a sense of  $u \in V$ 
15:      $\widehat{\text{ctx}}(\hat{u}) \leftarrow \widehat{\text{ctx}}(\hat{u}) \cup \{\hat{v}\}$ 
16:    $\mathcal{E} \leftarrow \{\{\hat{u}, \hat{v}\} \in \mathcal{V}^2 : \hat{v} \in \widehat{\text{ctx}}(\hat{u})\}$            $\triangleright$  Global Step: Sense Graph Edges
17:    $\mathcal{G} \leftarrow (\mathcal{V}, \mathcal{E})$                        $\triangleright$  Global Step: Sense Graph Construction
18:    $\mathcal{C} \leftarrow \text{Cluster}_{\text{Global}}(\mathcal{G})$            $\triangleright$  Global Step: Sense Graph Clustering
19:    $C \leftarrow \{\{u \in V : \hat{u} \in \mathcal{C}^i\} \subseteq V : \mathcal{C}^i \in \mathcal{C}\}$            $\triangleright$  Remove the sense labels
20: return  $C$ 
```

Local Step: Node Sense Induction and Disambiguation

The *local* step of Watset discovers the node senses in the input graph and uses this information to discover which particular senses of the nodes were connected via the edges of the input graph G .

Node Sense Induction. We induce node senses using the word neighborhood clustering approach by [43]. In particular, we assume that the removal of the nodes participating in many triangles separates a graph into several connected components. Each component corresponds to the sense of the target node, so this procedure is executed for every node independently. Figure 2-2 illustrates this approach for sense induction. For related work on word sense induction approaches.

Given a node $u \in V$, we extract its open neighborhood $G_u = (V_u, E_u)$ from the input

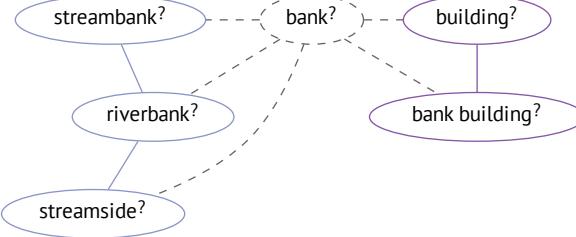


Figure 2-2: Clustering the neighborhood of the node “bank” of the input graph results in two clusters treated as the non-disambiguated sense contexts: $\text{bank}^1 = \{\text{streambank}, \text{riverbank}, \dots\}$ and $\{\text{bank}^2 = \text{bank building}, \text{building}, \dots\}$.

Table 2.1: Example of induced senses for the node “bank” and the corresponding clusters (contexts).

Sense	Context
bank^1	$\{\text{streambank}, \text{riverbank}, \dots\}$
bank^2	$\{\text{bank building}, \text{building}, \dots\}$
bank^3	$\{\text{bank company}, \dots\}$
bank^4	$\{\text{coin bank}, \text{penny bank}, \dots\}$

graph G , such that the target node u is not included into V_u (lines 3–5):

$$V_u = \{v \in V : \{u, v\} \in E\}, \quad (2.1)$$

$$E_u = \{\{v, w\} \in E : v, w \in V_u\}. \quad (2.2)$$

Then, we run a hard graph clustering algorithm on G_u that assigns one node to one and only one cluster, yielding a clustering C_u (line 6). We treat each obtained cluster $C_u^i \in C_u \subset V_u$ as representing a context for a different sense of the node $u \in V$ (lines 7–9). We denote, e.g., bank^1 , bank^2 and other labels as the node *senses* referred to as *senses(bank)*. In the example in Table 2.1, $|\text{senses}(\text{bank})| = 4$. Given a sense $u^i \in \text{senses}(u)$, we denote $\text{ctx}(u^i) = C_u^i$ as a *context* of this sense of the node $u \in V$. Execution of this procedure for all the words in V results in the set of senses for the global step (line 10):

$$\mathcal{V} = \bigcup_{u \in V} \text{senses}(u). \quad (2.3)$$

Disambiguation of Neighbors. Although at the previous step we have induced node senses and mapped them to the corresponding contexts (Table 2.1), the elements of these

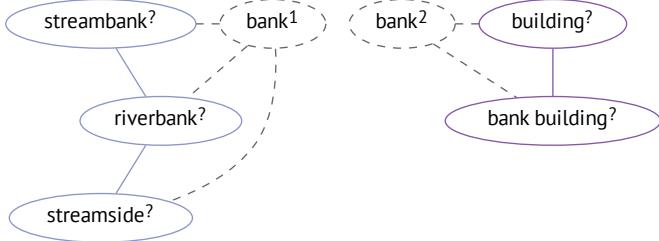


Figure 2-3: Contexts for two different senses of the node “bank”: only its senses bank^1 and bank^2 are currently known, while the other nodes in contexts need to be disambiguated.

Table 2.2: An example of context vectors for the node senses demonstrated in Figures 2-3 and 2-4. Since the graph is unweighted, one-hot encoding has been used. For matching purposes, the word “bank” is temporarily added into $\text{ctx}(\text{bank}^2)$.

Sense	bank	bank building	building	construction	edifice
bank^2	1	1	1	0	0
building^1	1	1	0	1	0
building^2	0	0	0	0	1

contexts do not contain sense information. For example, the context of bank^2 in Figure 2-3 has two elements $\{\text{bank building?}, \text{building?}\}$, the sense labels of which are currently not known. We recover the sense labels of nodes in a context using the sense disambiguated as follows.

We represent each context as a vector in a vector space model [44] constructed for all the contexts. Since the graph G is simple and the context of any sense $\hat{u} \in \mathcal{V}$ does not include the corresponding node $u \in V$ (Table 2.1), we *temporarily* put it into the context during disambiguation. This prevents the situation of non-matching when the context of a candidate sense $v' \in \text{senses}(v)$ has only one element and that element is u , i.e., $\text{ctx}(v') = \{u\}$. We intentionally perform this insertion temporarily only during matching to prevent self-referencing. When a context $\text{ctx}(\hat{u}) \subset V$ is transformed into a vector, we assign to each element $v \in \text{ctx}(\hat{u})$ of this vector a weight equal to the weight of the edge $\{u, v\} \in E$ of the input graph G . If G is unweighted, we assign 1 if and only if $\{u, v\} \in E$, otherwise 0 is assigned. Table 2.2 shows an example of the context vectors used for disambiguating the word *building* in the context of the sense bank^2 in Figure 2-3. In this example the vectors essentially represent one-hot encoding as the example input graph is unweighted.

Then, given a sense $\hat{u} \in \mathcal{V}$ of a node $u \in V$ and the context of this sense $\text{ctx}(\hat{u}) \subset V$, we

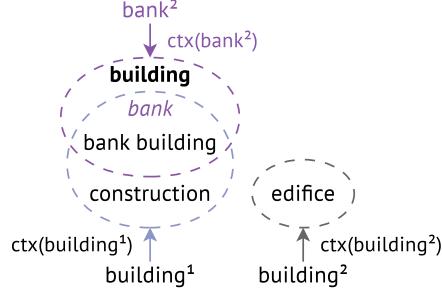


Figure 2-4: Matching the meaning of the ambiguous node “building” in the context of the sense bank^2 . For matching purposes, the word “bank” is temporarily added into $\text{ctx}(\text{bank}^2)$.

disambiguate each node $v \in \text{ctx}(\hat{u})$. For that, we find the sense $\hat{v} \in \text{senses}(v)$ the context $\text{ctx}(\hat{v}) \subset V$ of which maximizes the similarity to the target context $\text{ctx}(\hat{u})$. We compute the similarity using a context similarity measure $\text{sim} : (\text{ctx}(a), \text{ctx}(b)) \rightarrow \mathbb{R}$, $\forall \text{ctx}(a), \text{ctx}(b) \subseteq V$. Typical choices for the similarity measure are dot product, cosine similarity, Jaccard index, etc. Hence, we *disambiguate* each context element $v \in \text{ctx}(\hat{u})$:

$$\hat{v} = \arg \max_{v' \in \text{senses}(v)} \text{sim}(\text{ctx}(\hat{u}) \cup \{u\}, \text{ctx}(v')). \quad (2.4)$$

An example in Figure 2-4 illustrates the node sense disambiguation process. The context of the sense bank^2 is $\text{ctx}(\text{bank}^2) = \{\text{building}, \text{bank building}\}$ and the disambiguation target is *building*. Having chosen cosine similarity as the context similarity measure, we compute the similarity between $\text{ctx}(\text{bank}^2) \cup \{\text{bank}\}$ and the context of every sense of *building* in Table 2.2: $\cos(\text{ctx}(\text{bank}^2) \cup \{\text{bank}\}, \text{ctx}(\text{building}^1)) = \frac{2}{3}$ and $\cos(\text{ctx}(\text{bank}^2) \cup \{\text{bank}\}, \text{ctx}(\text{building}^2)) = 0$. Therefore, for the word *building* in the context of bank^2 , its first sense, building^1 , should be used because its similarity value is higher.

Finally, we construct a disambiguated context $\widehat{\text{ctx}}(\hat{u}) \subset \mathcal{V}$ which is a sense-aware representation of $\text{ctx}(\hat{u})$. This disambiguated context indicates which node senses were connected to $\hat{u} \in \mathcal{V}$ in the input graph G . For that, in lines 13–15, we apply the disambiguation procedure defined in Equation (2.4) for every node $v \in \text{ctx}(\hat{u})$:

$$\widehat{\text{ctx}}(\hat{u}) = \{\hat{v} \in \mathcal{V} : v \in \text{ctx}(\hat{u})\}. \quad (2.5)$$

As the result of the *local* step, for each node $u \in V$ in the input graph, we induce

the senses(u) $\subset \mathcal{V}$ of nodes and provide each sense $\hat{u} \in \mathcal{V}$ with a disambiguated context $\widehat{\text{ctx}}(\hat{u}) \subseteq \mathcal{V}$.

Global Step: Sense Graph Construction and Clustering

The *global* step of Watset constructs an intermediate *sense graph* expressing the connections between the node senses discovered at the *local* step. We assume that the nodes \mathcal{V} of the sense graph are non-ambiguous, so running a hard clustering algorithm on this graph outputs clusters C covering the set of nodes V of the input graph G .

Sense Graph Construction. Using the set of node senses defined in Equation (2.3), we construct the sense graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ by establishing undirected edges between the senses connected through the disambiguated contexts (lines 16–17):

$$\mathcal{E} = \{\{\hat{u}, \hat{v}\} \in \mathcal{V}^2 : \hat{v} \in \widehat{\text{ctx}}(\hat{u})\}. \quad (2.6)$$

Note that this edge construction approach disambiguates the edges E such that if a pair of nodes was connected in the input graph G , then the corresponding sense nodes will be connected in the sense graph \mathcal{G} . As the result, the constructed sense graph \mathcal{G} is a sense-aware representation of the input graph G . In case G is weighted, we assign each edge $\{\hat{u}, \hat{v}\} \in \mathcal{E}$ the same weight as the edge $\{u, v\} \in E$ has in the input graph.

Sense Graph Clustering. Running a hard clustering algorithm on \mathcal{G} produces the set of sense-aware clusters \mathcal{C} , each sense-aware cluster $\mathcal{C}^i \in \mathcal{C}$ is a subset of \mathcal{V} (line 18). In order to obtain the set of clusters C that covers the set of nodes V of the input graph G , we simply remove the sense labels from the elements of clusters \mathcal{C} (line 19):

$$C = \{\{u \in V : \hat{u} \in \mathcal{C}^i\} \subseteq V : \mathcal{C}^i \in \mathcal{C}\}. \quad (2.7)$$

Figure 2-5 illustrates the sense graph and its clustering on the example of the node “bank”. The construction of a sense graph requires disambiguation of the input graph nodes. Note that traditional approaches to graph-based sense induction, such as the ones proposed by [45–47], do not perform this step, but perform only local clustering of the graph since they do not aim at a global representation of clusters.

As the result of the *global* step, a set of clusters C of the input graph G is obtained using an intermediate sense-aware graph \mathcal{G} . The presented local-global graph clustering approach, Watset, makes it possible to naturally achieve a *soft* clustering of a graph using

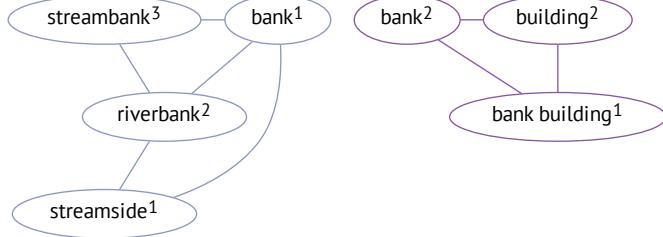


Figure 2-5: Clustering of the *sense graph* \mathcal{G} yields two clusters, $\{\text{bank}^1, \text{streambank}^3, \text{riverbank}^2, \dots\}$ and $\{\text{bank}^2, \text{bank building}^1, \text{building}^2, \dots\}$; if one removes the sense labels, the clusters will overlap resulting in a *soft* clustering of the input graph G .

hard clustering algorithms only.

Simplified Watset

The original Watset algorithm described above has context construction and disambiguation steps. These steps involve computation of a context similarity measure, which needs to be chosen as a hyper-parameter of the algorithm. In this section, we describe a simplified version of Watset (Algorithm 2) that requires no context similarity measure, which leads to faster computation in practice with less hyper-parameter tuning.¹ As our experiments show, this simplified version demonstrates similar performance to the original Watset algorithm.

In the input graph G a pair of nodes $\{u, v\} \in V^2$ can be incident to one and only one edge. Otherwise these nodes are not connected. Due to the use of a *hard* clustering algorithm for node sense induction, in any pair of nodes $\{u, v\} \in E$, the node v can appear in the context of only one sense of u and vice versa. Therefore, we can omit the context disambiguation step by tracking the node sense identifiers produced during sense induction.

Given a pair $\{u, v\} \in E$, we reuse the sense information from Table 2.1 to determine which context of a sense $\hat{u} \in \mathcal{V}$ contains v . We denote this as $\text{senses}[u][v] \in \mathbb{N}$, which indicates $v \in \text{ctx}(u^{\text{senses}[u][v]})$, i.e., the fact that node v is connected to the node u in the specified sense $u^{\text{senses}[u][v]}$. Following the example in Figure 2-2, if the context of bank^1 contains the word *streambank* then the context of one of the senses of *streambank* must contain the word *bank*, e.g., *streambank*³. This information allows us to create Table 2.3 that allows producing the set of sense-aware edges by simultaneously retrieving the corresponding

¹The simplified version of Watset was proposed by Dmitry Ustalov and is not part of contributions, but we present it here for completeness.

Algorithm 2 Simplified Watset.

Input: graph $G = (V, E)$, hard clustering algorithms $\text{Cluster}_{\text{Local}}$ and $\text{Cluster}_{\text{Global}}$.
Output: clusters C .

```

1:  $\mathcal{V} \leftarrow \emptyset$ 
2: for all  $u \in V$  do ▷ Local Step: Sense Induction
3:    $V_u \leftarrow \{v \in V : \{u, v\} \in E\}$  ▷ Note that  $u \notin V_u$ 
4:    $E_u \leftarrow \{\{v, w\} \in E : v, w \in V_u\}$ 
5:    $G_u \leftarrow (V_u, E_u)$ 
6:    $C_u \leftarrow \text{Cluster}_{\text{Local}}(G_u)$  ▷ Cluster the open neighborhood of  $u$ 
7:   for all  $C_u^i \in C_u$  do
8:     for all  $v \in C_u^i$  do ▷ Node  $v$  is connected to the  $i$ -th sense of  $u$ 
9:        $\text{senses}[u][v] \leftarrow i$ 
10:       $\mathcal{V} \leftarrow \mathcal{V} \cup \{u^i\}$ 
11:       $\mathcal{E} \leftarrow \{\{u^{\text{senses}[u][v]}, v^{\text{senses}[v][u]}\} \in \mathcal{V}^2 : \{u, v\} \in E\}$  ▷ Global Step: Sense Graph Edges
12:       $\mathcal{G} \leftarrow (\mathcal{V}, \mathcal{E})$  ▷ Global Step: Sense Graph Construction
13:       $\mathcal{C} \leftarrow \text{Cluster}_{\text{Global}}(\mathcal{G})$  ▷ Global Step: Sense Graph Clustering
14:       $C \leftarrow \{\{u \in V : \hat{u} \in \mathcal{C}^i\} \subseteq V : \mathcal{C}^i \in \mathcal{C}\}$  ▷ Remove the sense labels
15: return  $C$ 

```

sense identifiers:

$$\mathcal{E} = \left\{ \{u^{\text{senses}[u][v]}, v^{\text{senses}[v][u]}\} \in \mathcal{V}^2 : \{u, v\} \in E \right\}. \quad (2.8)$$

This allows us to construct the sense graph \mathcal{G} in linear time $O(|E|)$ by querying the node sense index to disambiguate the input edges E in a deterministic way. Other steps are identical to the original Watset algorithm. Simplified Watset is presented in Algorithm 2.

2.3 Results

We demonstrate that algorithm shows competitive results in three applications: unsupervised synset induction from a synonymy graph, unsupervised semantic frame induction from dependency triples, and unsupervised semantic class induction from a distributional thesaurus. Our algorithm is generic and can be also applied to other networks of linguistic data (see Table 2.4). Further details, including theoretical and experimental algorithmic complexity analysis and applications to real graphs can be found in [6, 9, 10] and Appendices A.6, A.9, and A.10

Table 2.3: Node sense identifier tracking in Simplified Watset as according to Figure 2-2.

Source	Target	Index
bank	streambank	1
	riverbank	1
	streamside	1
	building	2
	bank building	2
streambank	bank	3
	riverbank	3
...		

Table 2.4: Various types of input linguistic graphs clustered by the Watset algorithm and the corresponding induced output symbolic linguistic structures.

Input Nodes	Input Edges	Output Linguistic Structure
Polysemous words	Synonymy relationships	Synsets composed of disambiguated words
Subject-Verb-Object (SVO) triples	Most distributionally similar SVO triples	Lexical semantic frames
Polysemous words	Most distributionally similar words	Semantic classes composed of disambiguated words

Chapter 3

Word Sense Embeddings

Materials of this chapter are based on papers [17, 20] from the list of 14 publications the thesis is based on. Full texts are available in Appendix A.11 and A.14.

3.1 Introduction

In this chapter, graph clustering is applied to lexical-semantic networks generated from vector representations of words to obtain vector representations of word senses. A simple yet effective approach for learning word sense embeddings is introduced. In contrast to existing techniques, which either directly learn sense representations from corpora or rely on sense inventories from lexical resources, our approach can induce a sense inventory from existing word embeddings via clustering of ego-networks of related words. An integrated WSD mechanism enables labelling of words in context with learned sense vectors, which gives rise to downstream applications.

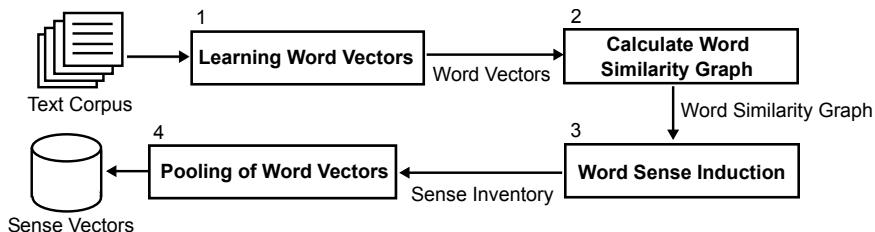


Figure 3-1: Schema of the word sense embeddings learning method “SenseGram”.

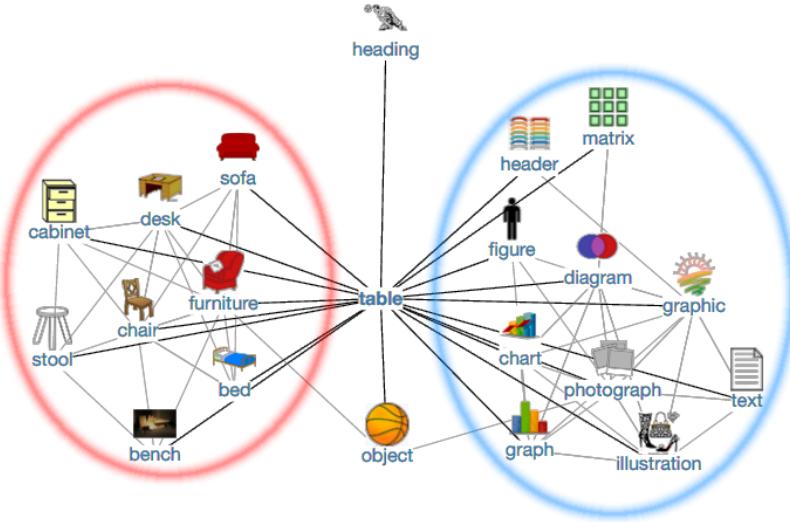


Figure 3-2: Visualization of the ego-network of “table” with furniture and data sense clusters. Note that the target “table” is excluded from clustering.

3.2 Method

Our method consists of the four main stages depicted in Figure 3-1: (1) learning word embeddings; (2) building a graph of nearest neighbours based on vector similarities; (3) induction of word senses using ego-network clustering; and (4) aggregation of word vectors with respect to the induced senses.

Our method can use existing word embeddings, sense inventories and word similarity graphs. To demonstrate such use-cases and to study the performance of the method in different settings, as variants of the complete pipeline presented in Figure 3-1, we experiment with two additional setups. The sense vectors are then constructed by averaging embeddings of words in each resulting cluster. In order to use these sense vectors for word sense disambiguation in text, the authors compute the probabilities of sense vectors of a word given its context or the similarity of the sense vectors to the context. Below we describe each of the stages of our method in detail.

Learning Word Vectors

To learn word vectors, we use the *word2vec* [48] and fastText [49] but similar pre-trained word embeddings can be used. The final sense embeddings remain in the same vector space as these input word vectors.

Calculating Word Similarity Graph

At this step, we build a graph of word similarities, such as (table, desk, 0.78). For each word we retrieve its 200 nearest neighbours. This number is motivated by prior studies [50, 51]: as observed, only few words have more strongly semantically related words. This graph is computed either based on word embeddings learned during the previous step or using semantic similarities provided by the *JoBimText* framework [50].

For similarities using word embeddings, such as word2vec, nearest neighbours of a term are terms with the highest cosine similarity of their respective vectors. For similarities using JoBimText (GBT) every word is represented as a bag of sparse dependency-based features extracted using the Malt parser and collapsed using an approach similar to [52]. Features are normalized using the LMI score [53] and further pruned down. Similarity of two words is equal to the number of common features.

Algorithm 3 Word sense induction: Baseline Algorithm.

```
input :  $T$  – word similarity graph,  $N$  – ego-network size,  $n$  – ego-network connectivity,  $k$ 
       – minimum cluster size
output: for each term  $t \in T$ , a clustering  $S_t$  of its  $N$  most similar terms
1 foreach  $t \in T$  do
2    $V \leftarrow N$  most similar terms of  $t$  from  $T$ 
     $G \leftarrow$  graph with  $V$  as nodes and no edges  $E$ 
3   foreach  $v \in V$  do
4      $V' \leftarrow n$  most similar terms of  $v$  from  $T$ 
      foreach  $v' \in V'$  do
        | if  $v' \in V$  then add edge  $(v, v')$  to  $E$ 
5     end
6   end
7
8    $S_t \leftarrow \text{ChineseWhispers}(G)$ 
    $S_t \leftarrow \{s \in S_t : |s| \geq k\}$ 
9 end
```

Word Sense Induction: a Baseline Algorithm

A word sense is represented by a word cluster. For instance the cluster “chair, bed, bench, stool, sofa, desk, cabinet” can represent the sense “table (furniture)”. To induce senses, first we construct an ego-network G of a word t and then perform graph clustering of this network. The identified clusters are interpreted as senses (see Figure 3-2). Words referring to the same sense tend to be tightly connected, while having fewer connections to words referring to different senses.

The sense induction presented in Algorithm 3 processes one word t of the word similarity graph T per iteration. First, we retrieve nodes V of the ego-network G : these are the N most similar words of t according to T . The target word t itself is not part of the ego-network. Second, we connect the nodes in G to their n most similar words from T . Finally, the ego-network is clustered with the Chinese Whispers algorithm [54]. This method is parameter free, thus we make no assumptions about the number of word senses.

The sense induction algorithm has three meta-parameters: the ego-network size (N) of the target ego word t ; the ego-network connectivity (n) is the maximum number of connections the neighbour v is allowed to have within the ego-network; the minimum size of the cluster k . The n parameter regulates the granularity of the inventory. In our experiments, we set the N to 200, n to 50, 100 or 200 and k to 5 or 15 to obtain different granulates, cf. [55]. Each word in a sense cluster has a weight which is equal to the similarity score between this word and the ambiguous word t .

Word Sense Induction: an Improved Algorithm

One of the downsides of the described above algorithm is noise in the generated graph, namely, unrelated words and wrong connections. They hamper the separation of the graph. Another weak point is the imbalance in the nearest neighbour list, when a large part of it is attributed to the most frequent sense, not sufficiently representing the other senses. This can lead to construction of incorrect sense vectors.

We suggest an improved procedure of graph construction that uses the interpretability of vector addition and subtraction operations in word embedding space [56] while the previous algorithm only relies on the list of nearest neighbours in word embedding space. The key innovation is the use of vector subtraction to find pairs of most dissimilar graph nodes and construct the graph only from the nodes included in such “anti-edges”. Thus, our algorithm is based on *graph-based* word sense induction, but it also relies on *vector-based* operations between word embeddings to perform filtering of graph nodes. Analogously to the method above, we construct a semantic relatedness graph from a list of nearest neighbours, but we filter this list using the following procedure:

1. Extract a list $\mathcal{N} = \{w_1, w_2, \dots, w_N\}$ of N nearest neighbours for the target (ego) word vector w .
2. Compute a list $\Delta = \{\delta_1, \delta_2, \dots, \delta_N\}$ for each w_i in \mathcal{N} , where $\delta_i = w - w_i$. The vectors

in δ contain the components of sense of w which are not related to the corresponding nearest neighbours from \mathcal{N} .

3. Compute a list $\overline{\mathcal{N}} = \{\overline{w_1}, \overline{w_2}, \dots, \overline{w_N}\}$, such that $\overline{w_i}$ is in the top nearest neighbours of δ_i in the embedding space. In other words, $\overline{w_i}$ is a word which is the most similar to the target (ego) word w and least similar to its neighbour w_i . We refer to $\overline{w_i}$ as an *anti-node* of w_i . The set of N nearest neighbours and their anti-nodes form a set of *anti-edges* i.e. pairs of most dissimilar nodes – those which should not be connected: $\overline{E} = \{(w_1, \overline{w_1}), (w_2, \overline{w_2}), \dots, (w_N, \overline{w_N})\}$.

To clarify this, consider the target (ego) word $w = \text{python}$, its top similar term $w_1 = \text{Java}$ and the resulting anti-node $\overline{w_1} = \text{snake}$ which is the top related term of $\delta_1 = w - w_1$. Together they form an anti-edge $(w_1, \overline{w_1}) = (\text{Java}, \text{snake})$ composed of a pair of semantically dissimilar terms.

4. Construct V , the set of vertices of our semantic graph $G = (V, E)$ from the list of anti-edges \overline{E} , with the following recurrent procedure: $V = V \cup \{w_i, \overline{w_i} : w_i \in \mathcal{N}, \overline{w_i} \in \mathcal{N}\}$, i.e. we add a word from the list of nearest neighbours *and* its anti-node only if both of them are nearest neighbours of the original word w . We do not add w 's nearest neighbours if their anti-nodes do not belong to \mathcal{N} . Thus, we add only words which can help discriminating between different senses of w .
5. Construct the set of edges E as follows. For each $w_i \in \mathcal{N}$ we extract a set of its K nearest neighbours $\mathcal{N}'_i = \{u_1, u_2, \dots, u_K\}$ and define $E = \{(w_i, u_j) : w_i \in V, u_j \in V, u_j \in \mathcal{N}'_i, u_j \neq \overline{w_i}\}$. In other words, we remove edges between a word w_i and its nearest neighbour u_j if u_j is also its anti-node. According to our hypothesis, w_i and $\overline{w_i}$ belong to different senses of w , so they should not be connected (i.e. we never add anti-edges into E). Therefore, we consider any connection between them as noise and remove it.

Note that N (the number of nearest neighbours for the target word w) and K (the number of nearest neighbours of w_{ci}) do not have to match. The difference between these parameters is the following. N defines how many words will be considered for the construction of ego-graph. On the other hand, K defines the degree of relatedness between words in the ego-graph — if $K = 50$, then we will connect vertices w and u with an edge

only if u is in the list of 50 nearest neighbours of w . Increasing K increases the graph connectivity and leads to lower granularity of senses.

The described vertices selection procedure allows picking the most representative members of these clusters which are better at discriminating between the clusters. In addition to that, it helps dealing with the cases when one of the clusters is over-represented in the nearest neighbour list. In this case, many elements of such a cluster are not added to V because their anti-nodes fall outside the nearest neighbour list. This also improves the quality of clustering.

After the graph construction, the clustering is performed using the Chinese Whispers algorithm [46].

Figure 3-3 shows an example of the resulting pruned graph of for the word *Ruby* for $N = 50$ nearest neighbours in terms of the fastText cosine similarity. In contrast to the baseline method described above where all 50 terms are clustered, in the method presented in this section we sparsify the graph by removing 13 nodes which were not in the set of the “anti-edges” i.e. pairs of most dissimilar terms out of these 50 neighbours. Examples of anti-edges i.e. pairs of most dissimilar terms for this graph include: (*Haskell, Sapphire*), (*Garnet, Rails*), (*Opal, Rubyist*), (*Hazel, RubyOnRails*), and (*Coffeescript, Opal*).

Pooling of Word Vectors

At this stage, we calculate sense embeddings for each sense in the induced inventory. We assume that a word sense is a composition of words that represent the sense. We define a sense vector as a function of word vectors representing cluster items. Let W be a set of all words in the training corpus and let $S_i = \{w_1, \dots, w_n\} \subseteq W$ be a sense cluster obtained during the previous step. Consider a function $\text{vec}_w : W \rightarrow \mathbb{R}^m$ that maps words to their vectors and a function $\gamma_i : W \rightarrow \mathbb{R}$ that maps cluster words to their weight in the cluster S_i . We experimented with two ways to calculate sense vectors: unweighted average of word vectors:

$$\mathbf{s}_i = \frac{\sum_{k=1}^n \text{vec}_w(w_k)}{n}; \quad (3.1)$$

and weighted average of word vectors:

$$\mathbf{s}_i = \frac{\sum_{k=1}^n \gamma_i(w_k) \text{vec}_w(w_k)}{\sum_{k=1}^n \gamma_i(w_k)}. \quad (3.2)$$

Table 3.1 provides an example of weighted pooling results. While the original neighbours

Vector	Nearest Neighbours
table	tray, bottom, diagram, bucket, brackets, stack, basket, list, parenthesis, cup, trays, pile, playfield, bracket, pot, drop-down, cue, plate
table#0	leftmost#0, column#1, randomly#0, tableau#1, top-left#0, indent#1, bracket#3, pointer#0, footer#1, cursor#1, diagram#0, grid#0
table#1	pile#1, stool#1, tray#0, basket#0, bowl#1, bucket#0, box#0, cage#0, saucer#3, mirror#1, birdcage#0, hole#0, pan#1, lid#0

Table 3.1: Neighbours of the word “table” and its senses produced by our method. The neighbours of the initial vector belong to both senses, while those of sense vectors are sense-specific.

of the word “table” contain words related to both furniture and data, the neighbours of the sense vectors are either related to furniture or data, but not to both at the same time. Besides, each neighbour of a sense vector has a sense identifier as we calculate cosine between sense vectors, not word vectors.

Word Sense Disambiguation

This section describes how sense vectors are used to disambiguate a word in a context. Given a target word w and its context words $C = \{c_1, \dots, c_k\}$, we first map w to a set of its sense vectors according to the inventory: $S = \{\mathbf{s}_1, \dots, \mathbf{s}_n\}$. We use two strategies to choose a correct sense taking vectors for context words either from the matrix of context embeddings or from the matrix of word vectors. The first one is based on sense probability in given context:

$$s^* = \arg \max_i P(C|\mathbf{s}_i) = \arg \max_i \frac{1}{1 + e^{-\bar{\mathbf{c}}_c \cdot \mathbf{s}_i}}, \quad (3.3)$$

where $\bar{\mathbf{c}}_c$ is the mean of context embeddings: $k^{-1} \sum_{i=1}^k vec_c(c_i)$ and functions $vec_c : W \rightarrow \mathbb{R}^m$ map context words to context embeddings. Using the mean of context embeddings to calculate sense probability is natural with the CBOW because this model optimizes exactly the same mean to have high scalar product with word embeddings for words occurred in context and low scalar product for random words [48].

The second disambiguation strategy is based on similarity between sense and context:

$$s^* = \arg \max_i sim(\mathbf{s}_i, C) = \arg \max_i \frac{\bar{\mathbf{c}}_w \cdot \mathbf{s}_i}{\|\bar{\mathbf{c}}_w\| \cdot \|\mathbf{s}_i\|}, \quad (3.4)$$

where $\bar{\mathbf{c}}_w$ is the mean of word embeddings: $\bar{\mathbf{c}}_w = k^{-1} \sum_{i=1}^k vec_w(c_i)$. The latter method uses only word vectors (vec_w) and require no context vectors (vec_c). This is practical, as the standard implementation of *word2vec* does not save context embeddings and thus most pre-computed models provide only word vectors.

To improve WSD performance we also apply context filtering. Typically, only several words in context are relevant for sense disambiguation, like “chairs” and “kitchen” are for “table” in “They bought a table and chairs for kitchen.” For each word c_j in context $C = \{c_1, \dots, c_k\}$ we calculate a score that quantifies how well it discriminates the senses:

$$\max_i f(\mathbf{s}_i, c_j) - \min_i f(\mathbf{s}_i, c_j), \quad (3.5)$$

where \mathbf{s}_i iterates over senses of the ambiguous word and f is one of our disambiguation strategies: either $P(c_j|\mathbf{s}_i)$ or $sim(\mathbf{s}_i, c_j)$. The p most discriminative context words are used for disambiguation.

Knowledge-Free Labelling of Induced Senses

We label each word cluster representing a sense to make them and the WSD results interpretable by humans. In the chapter below we show how hypernyms can be used label the clusters, e.g. “animal” in the “python (animal)”. Yet hypernyms are not available for some low-resourced languages. Therefore, we describe a simpler method to select a keyword which would help to interpret each cluster. For each graph node $v \in V$ we count the number of anti-edges it belongs to:

$$keyness(v) = |\{(w_i, \overline{w_i}) : (w_i, \overline{w_i}) \in \overline{E} \wedge (v = w_i \vee v = \overline{w_i})\}|. \quad (3.6)$$

A graph clustering yields a partition of V into n clusters: $V = \{V_1, V_2, \dots, V_n\}$. For each cluster V_i we define a *keyword* w_i^{key} as the word with the largest number of anti-edges $keyness(\cdot)$ among words in this cluster.

3.3 Results

Main experiments were conducted for English language. Besides, We used this method to induce a collection of sense inventories for 158 languages on the basis of the original

pre-trained fastText word embeddings by [57], enabling WSD in these languages.

The results suggest that, the presented algorithms on English WSI datasets and multilingual lexical similarity and relatedness task show that the performance of our method is comparable to state-of-the-art unsupervised WSD systems.

Further experimental results and their analysis can be found in [17, 20] presented in Appendix A.11 and A.14.

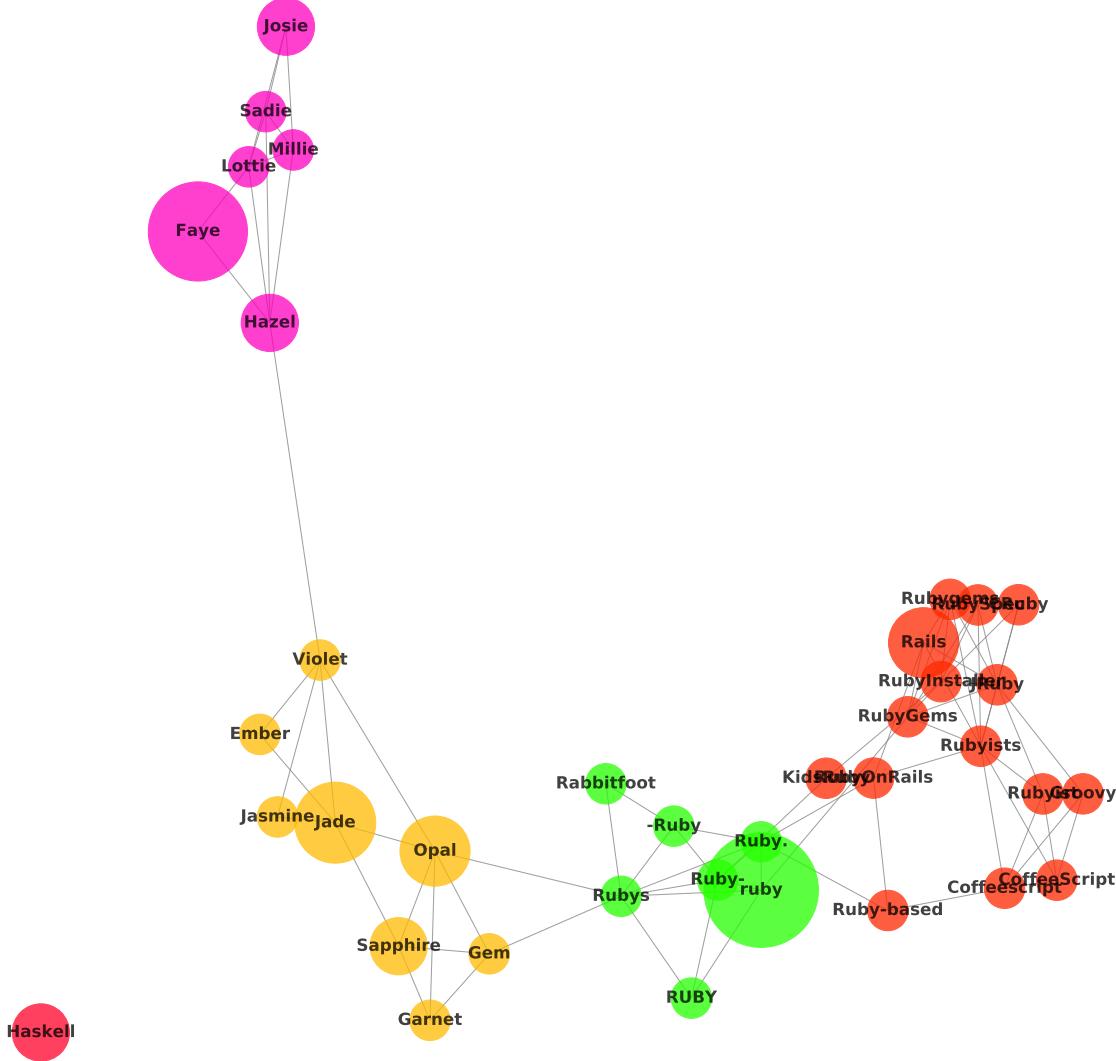


Figure 3-3: The graph of nearest neighbours of the word *Ruby* separated according several senses: programming languages, female names, gems, different spellings of the word *Ruby*. Node size denotes word importance with the largest node in the cluster being used as a keyword to interpret an induced word sense.

Chapter 4

Unsupervised Interpretable Word Sense Disambiguation

Materials of this chapter are based on papers [1] and [18] from the list of 14 publications the thesis is based on. Full texts are available in Appendices A.1 and A.12.

4.1 Introduction

In this chapter, an unsupervised, knowledge-free, and interpretable approach to word sense induction and disambiguation is proposed building on top of techniques presented in the previous two chapters.

The current trend in NLP is the use of highly opaque models, e.g. neural networks and word embeddings. While these models yield state-of-the-art results on a range of tasks, their drawback is poor interpretability. On the example of word sense induction and disambiguation (WSID), we show that it is possible to develop an interpretable model that matches the state-of-the-art models in accuracy. Namely, we present an unsupervised, knowledge-free WSID approach, which is interpretable at three levels: word sense inventory, sense feature representations, and disambiguation procedure. Experiments show that our model performs on par with state-of-the-art word sense embeddings and other unsupervised systems while offering the possibility to justify its decisions in human-readable form.

A word sense disambiguation (WSD) system takes as input a target word t and its context C . The system returns an identifier of a word sense s_i from the word sense

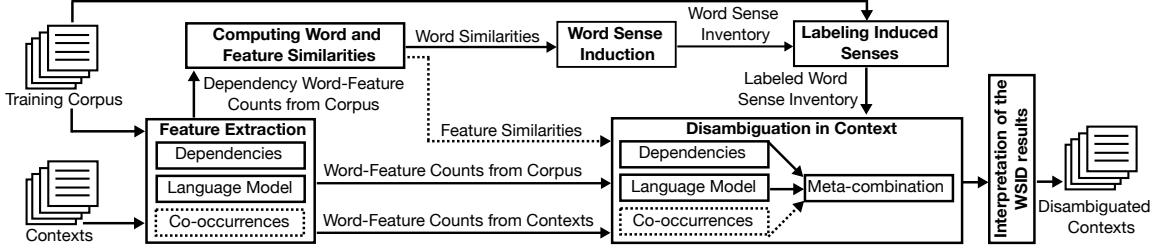


Figure 4-1: Outline of unsupervised interpretable method for word sense induction and disambiguation.

inventory $\{s_1, \dots, s_n\}$ of t , where the senses are typically defined manually in advance. Despite significant progress in methodology during the two last decades [58–60], WSD is still not widespread in applications [61], which indicates the need for further progress. The difficulty of the problem largely stems from the lack of domain-specific training data. A fixed sense inventory, such as the one of WordNet [62], may contain irrelevant senses for the given application and at the same time lack relevant domain-specific senses. Word sense induction from domain-specific corpora is supposed to solve this problem. However, most approaches to word sense induction and disambiguation, e.g. [63–65], rely on clustering methods and dense vector representations that make a WSD model uninterpretable as compared to knowledge-based WSD methods.

Interpretability of a statistical model is important as it lets us understand the reasons behind its predictions [66–68]. Interpretability of WSD models (1) lets a user understand why in the given context one observed a given sense (e.g., for educational applications); (2) performs a comprehensive analysis of correct and erroneous predictions, giving rise to improved disambiguation models.

The contribution of this chapter is an interpretable unsupervised knowledge-free WSD method. The novelty of our method is in (1) a technique to disambiguation that relies on induced inventories as a pivot for learning sense feature representations, (2) a technique for making induced sense representations interpretable by labeling them with hypernyms and images.

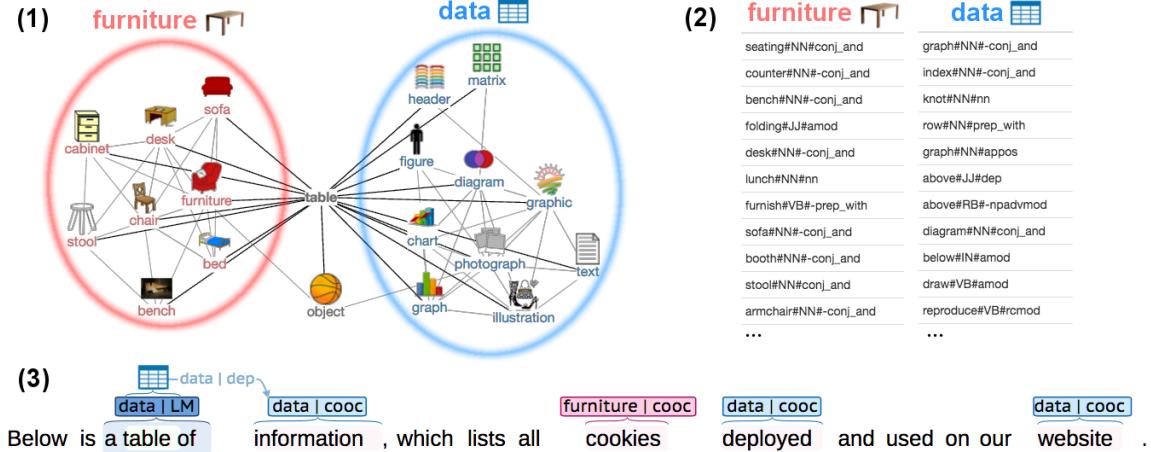


Figure 4-2: Interpretation of the senses of the word “table” at three levels by our method: (1) word sense inventory; (2) sense feature representation; (3) results of disambiguation in context. The sense labels (“furniture” and “data”) are obtained automatically based on cluster labeling with hypernyms. The images associated with the senses are retrieved using a search engine: “table data” and “table furniture”.

4.2 Method

Our unsupervised word sense disambiguation method consist of the five steps illustrated in Figure 4-1: extraction of context features (Section 3.1); computing word and feature similarities; word sense induction; labeling of clusters with hypernyms and images, disambiguation of words in context based on the induced inventory, and finally interpretation of the model. Feature similarity and co-occurrence computation steps (drawn with a dashed lines) are optional, since they did not consistently improve performance.

Extraction of Context Features The goal of this step is to extract word-feature counts from the input corpus. In particular, we extract features based on dependencies, word co-occurrences, and statistical language models. Also graph of word and feature similarities are computed and used.

Word Sense Induction We induce a sense inventory by clustering of ego-network of similar words. In our case, an inventory represents senses by a word cluster, such as “chair, bed, bench, stool, sofa, desk, cabinet” for the “furniture” sense of the word “table”. The sense induction processes one word t of the distributional thesaurus T per iteration. First, we retrieve nodes of the ego-network G of t being the N most similar words of t according

to T (see Figure 4-2 (1)). Note that the target word t itself is not part of the ego-network. Second, we connect each node in G to its n most similar words according to T . Finally, the ego-network is clustered with Chinese Whispers [54]. The n parameter regulates the granularity of the inventory: we experiment with $n \in \{200, 100, 50\}$ and $N = 200$.

Labeling Induced Senses with Hypernyms and Images To improve interpretability of induced senses, we assign an image to each word in the cluster (see Figure 4-2) by querying the Bing image search API using the query composed of the target word and its hypernym, e.g. “jaguar car”. The first hit of this query is selected to represent the induced word sense.

Algorithm 4 Unsupervised WSD with induced word sense inventory.

```

input : Word  $t$ , context features  $C$ , sense inventory  $I$ , word-feature table  $F$ , use largest cluster
         back-off  $LCB$ , use feature expansion  $FE$ .
output: Sense of the target word  $t$  in inventory  $I$  and confidence score.

10  $S \leftarrow \text{getSenses}(I, t)$ 
   if  $FE$  then
11   |    $C \leftarrow \text{featureExpansion}(C)$ 
12 end
13 foreach  $(sense, cluster) \in S$  do
14   |    $\alpha[sense] \leftarrow \{\}$ 
15   |   foreach  $w \in cluster$  do
16     |   |   foreach  $c \in C$  do
17       |   |   |    $\alpha[sense] \leftarrow \alpha[sense] \cup F(w, c)$ 
18   |   end
19 end
20 if  $\max_{sense \in S} \text{mean}(\alpha[sense]) = 0$  then
21   |   if  $LCB$  then
22     |   |   return  $\arg \max_{(sense, cluster) \in S} |\alpha[sense]|$ 
23   |   else
24     |   |   return  $-1$  // reject to classify
25   end
26 else
27   |   return  $\arg \max_{(sense, cluster) \in S} \text{mean}(\alpha[sense])$ 
28 end

```

WSD with Induced Word Sense Inventory To disambiguate a target word t in context, we extract context features C and pass them to Algorithm 4. We use the induced sense inventory I and select the sense that has the largest weighted feature overlap with context features or fall back to the largest cluster back-off when context features C do not match the learned sense representations.

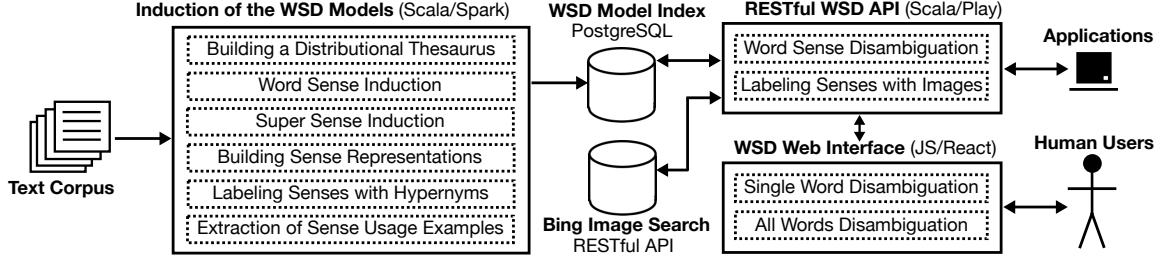


Figure 4-3: Software and functional architecture of the WSD system.

The algorithm starts by retrieving induced sense clusters of the target word (line 1). Next, the method starts to accumulate context feature weights of each *sense* in $\alpha[sense]$. Each word w in a sense *cluster* brings all its word-feature counts $F(w, c)$: see lines 5-12. Finally, a *sense* that maximizes mean weight across all context features is chosen (lines 13-21). Optionally, we can resort to the largest cluster back-off (LCB) strategy in case if no context features match sense representations.

4.3 Results

We use two lexical sample collections suitable for evaluation of unsupervised WSD systems: Turk Bootstrap Word Sense Inventory (TWSI) dataset introduced by [69] and SemEval 2013 word sense induction dataset by [70].

The presented method yields performance comparable to the state-of-the-art unsupervised systems, including two methods based on word sense embeddings. Note, however, that none of the rivalling systems has a comparable level of interpretability to our approach. Further details of experimental results can be found in [1] and [18] and Appendices A.1 and A.12.

In addition to experimental results, an open source implementation of the method featuring a web demo of several pre-trained models was released.¹ System architecture is illustrated in Figure 4-3: it features API and a web application with user interface for interpretable WSD and sense inventory navigation. The application performs human-interpretable disambiguation of a text entered by a user in single word disambiguation mode (cf. Figure 4-4) and all words disambiguation mode (cf. Figure 4-5).

¹<http://www.jobimtext.org/wsd>

Sentence
Jaguar is a large spotted predator of tropical America similar to the leopard. A

Word
Jaguar B

Model
Word Senses based on Cluster Word Features C

PREDICT SENSE RANDOM SAMPLE

Predicted senses for 'Jaguar'

 **1. jaguar (animal)**
Similarity score: 0.00184 / Confidence: 99.87% / Sense ID: jaguar#0 / BabelNet ID: bn:00033987n

Hypernyms animal wildlife bird mammal D

Sample sentences
The **jaguar**, a compact and well-muscled animal, is the largest cat in the New World.
Jaguar may leap onto the back of the prey and sever the cervical vertebrae, immobilizing the target.

Cluster words lion tiger leopard wolf monkey otter crocodile alligator deer cat elephant fox eagle owl snake E

Context words elephant: 0.012 tiger: 0.012 fox: 0.0099 wolf: 0.0097 cub: 0.0086 monkey: 0.0083 leopard: 0.0074 eagle: 0.0062 den: 0.0043 elk: 0.0040 32078 more not shown F

Matching features leopard: 0.0011 predator: 0.00040 spotted: 0.00038 large: 0.0000041 similar: 0.0000015 tropical: 5.6e-7 america: 2.0e-7 E

BABELNET LINK F SHOW LESS E

Figure 4-4: Single word disambiguation mode: disambiguation of the word “Jaguar” (B) in the sentence “*Jaguar* is a large spotted predator of tropical America similar to the leopard.” (A) using the WSD disambiguation model based on cluster word features (C). The predicted sense is summarized with a hypernym and an image (D) and further represented with usage examples, semantically related words, and typical context clues. Each of these elements is extracted automatically. The reasons of the predictions are provided in terms of common sparse features of the input sentence and a sense representation (E). The induced senses are linked to BabelNet (F).

Sentence
Jaguar is a large spotted predator of tropical America similar to the leopard. A

Model
Word Senses based on Cluster Word Features C

DISAMBIGUATE SENTENCE RANDOM SAMPLE

Detected Entities
The system has detected these entities in the given sentence.

 animal Jaguar D	 animal predator D	 country America D
is a large spotted	of tropical	

Figure 4-5: All words disambiguation mode: results of disambiguation of nouns.

Chapter 5

Linking Word Sense Representations

Materials of this chapter are based on papers [7,8] from the list of 14 publications the thesis is based on. Full texts are available in Appendices A.7 and A.8.

5.1 Introduction

In this chapter, methods for linking of word sense representations induced automatically from text using the methods presented in the previous chapters to lexical-semantic networks constructed manually, e.g. Wordnets, are presented. The linking is performed on the level of individual word senses.

We present an approach to combining distributional semantic representations induced from text corpora with manually constructed lexical-semantic networks. While both kinds of semantic resources are available with high lexical coverage, our aligned resource combines the domain specificity and availability of contextual information from distributional models with the conciseness and high quality of manually crafted lexical networks. We start with a distributional representation of induced senses of vocabulary terms, which are accompanied with rich context information given by related lexical items. We then automatically disambiguate such representations to obtain a full-fledged proto-conceptualization, i.e. a typed graph of induced word senses. In a final step, this proto-conceptualization is aligned to a lexical ontology, resulting in a hybrid aligned

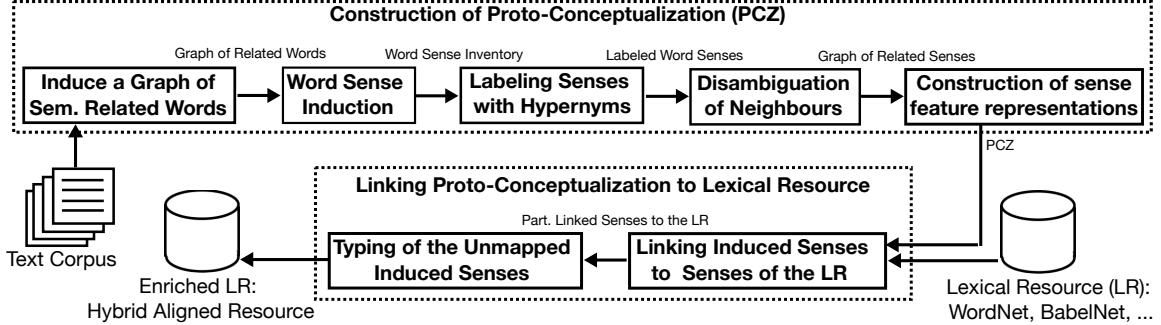


Figure 5-1: Overview of the proposed framework for enriching lexical resources: a distributional semantic model is used to construct a disambiguated distributional lexical semantic network (a proto-conceptualization, PCZ), which is subsequently linked to the lexical resource.

resource. Moreover, unmapped induced senses are associated with a semantic type in order to connect them to the core resource.

5.2 Method

The construction of our hybrid aligned resource (HAR) builds upon methods used to link various manually constructed lexical resources to construct BabelNet [71] and UBY [72], among others. In our method, however, linking is performed between two networks that are structurally similar, but have been constructed in two completely different ways: one resource is built using an unsupervised bottom-up approach from text corpora, while the second is constructed in a top-down manner using manual labor, e.g., codified knowledge from human experts such as lexicographers (WordNet). In particular, the method consists of two major phases, as illustrated in Figure 5-1.

The upper part correspond to the method described in the chapters above i.e. building interpretable word sense representations from text in an unsupervised manner. Below we describe how a corpus-induced semantic network (a proto-conceptualization) is linked to a manually created semantic network, represented by a lexical resource.

Linking Induced Senses to Senses of the Lexical Resource

We link each sense in our proto-conceptualization (PCZ) to the most suitable sense (if any) of a Lexical Resource (LR, see Figure 5-1 step 3). There exist many algorithms for knowledge base linking [73]: here, we build upon simple, yet high-performing previous approaches to linking LRs that achieved state-of-the-art performance. These rely at their

Algorithm 5 Linking induced senses to senses of a lexical resource.

Input: $T = \{(j_i, R_{j_i}, H_{j_i})\}$, W , th , m
Output: $M = (\text{source}, \text{target})$

- 1: $M = \emptyset$
- 2: **for all** $(j_i, R_{j_i}, H_{j_i}) \in T.\text{monosemousSenses}$ **do**
- 3: $C(j_i) = W.\text{getSenses}(j_i.\text{lemma}, j_i.\text{POS})$
- 4: **if** $|C(j_i)| == 1$, let $C(j_i) = \{c_0\}$ **then**
- 5: **if** $\text{sim}(j_i, c_0, \emptyset) \geq th$ **then**
- 6: $M = M \cup \{(j_i, c_0)\}$
- 7: **for** $\text{step} = 1$; $\text{step} \leq m$; $\text{step} = \text{step} + 1$ **do**
- 8: $M_{\text{step}} = \emptyset$
- 9: **for all** $(j_i, R_{j_i}, H_{j_i}) \in T.\text{senses}/M.\text{senses}$ **do**
- 10: $C(j_i) = W.\text{getSenses}(j_i.\text{lemma}, j_i.\text{POS})$
- 11: **for all** $c_k \in C(j_i)$ **do**
- 12: $\text{rank}(c_k) = \text{sim}(j_i, c_k, M)$
- 13: **if** $\text{rank}(c_k)$ has a single top value for c_t **then**
- 14: **if** $\text{rank}(c_t) \geq th$ **then**
- 15: $M_{\text{step}} = M_{\text{step}} \cup \{(j_i, c_t)\}$
- 16: $M = M \cup M_{\text{step}}$
- 17: **for all** $(j_i, R_{j_i}, H_{j_i}) \in T.\text{senses}/M.\text{senses}$ **do**
- 18: $M = M \cup \{(j_i, j_i)\}$
- 19: **return** M

core on computing the overlap between the bags of words built from the LRs' concept lexicalizations, e.g., [72, 74] (*inter alia*). Specifically, we develop i) an iterative approach – so that the linking can benefit from the availability of linked senses from previous iterations – ii) leveraging the lexical content of the source and target resources. Algorithm 5 takes as input:

1. a PCZ $T = \{(j_i, R_{j_i}, H_{j_i})\}$ where j_i is a sense identifier (i.e. mouse:1), R_{j_i} the set of its semantically related senses (i.e. $R_{j_i} = \{\text{keyboard:1}, \text{computer:0}, \dots\}$) and H_{j_i} the set of its hypernym senses (i.e. $H_{j_i} = \{\text{equipment:3}, \dots\}$);
2. a LR W : we experiment with: WordNet, a lexical database for English and BabelNet, a very large multilingual ‘encyclopedic dictionary’;
3. a threshold th over the similarity between pairs of concepts and a number m of iterations as a stopping criterion.

The algorithm outputs a mapping M , which consists of a set of pairs of the kind $(\text{source}, \text{target})$ where $\text{source} \in T.\text{senses}$ is a sense of the input PCZ T and

$target \in W.senses \cup source$ is the most suitable sense of W or $source$ when no such sense has been identified.

The algorithm starts by creating an empty mapping M (line 1). Then for each monosemous sense (e.g., Einstein:0 is the only sense in the PCZ for the term Einstein) it searches for a candidate monosemous sense (lines 2-6). If such monosemous candidate senses exist (line 4), we compare the two senses (line 5) with the following similarity function:

$$sim(j, c, M) = \frac{|T.Bow(j, M, W) \cap W.Bow(c)|}{|T.Bow(j, M, W)|}, \quad (5.1)$$

where

1. $T.Bow(j, M, W)$ is the set of words containing all the terms extracted from related/hypernym senses of j and all the terms extracted from the related/hypernym (i.e., already linked in M) synsets in W . For each synset from the LR, we use all synonyms and content words of the gloss.
2. $W.Bow(c)$ contains the synonyms and the gloss content words for the synset c and all the related synsets of c .

Then a new link pair (j_i, c_0) is added to M if the similarity score between j_i and c_0 meets or exceeds the threshold th (line 5). At this point, we collected a first set of disambiguated (monosemous) senses in M and start to iteratively disambiguate the remaining (polysemous) senses in T (lines 7-16). This iterative disambiguation process is similar to the one we described for the monosemous case (lines 2-6), with the main difference that, due to the polysemy of the candidates synsets, we instead use the similarity function to rank all candidate senses (lines 11-12) and select the top-ranked candidates for the mapping (lines 13-15). At the end of each iteration, we add all collected pairs to M (line 16). Finally, all unlinked j of T , i.e. induced senses that have no corresponding LR sense, are added to the mapping M (lines 17- 18).

Typing of the Unmapped Induced Senses

An approach based on the bag-of-words from concept lexicalizations has the advantage of being simple, as well as high performing as we show later in the evaluation – cf. also findings from [74]. However, there could be still PCZ senses that cannot be mapped to the target lexical resource, e.g., because of vocabulary mismatches, sparse concepts' lexicalizations, or because they are simply absent in the resource.

Algorithm 6 Typing of the unmapped induced senses.

Input: $M = (\text{source}, \text{target})$, W **Output:** $H = (\text{source}, \text{type})$

```
1:  $H = \emptyset$ 
2: for all  $(\text{source}, \text{target}) \in M$  do
3:   if  $\text{target} \notin W$  then
4:      $\text{Rank} = 0$ 
5:     for all  $\text{related} \in R_{\text{source}}$ ,  $\exists (\text{related}, \text{trelated}) \in M, \text{trelated} \in W$  do
6:       for all  $\text{hop} \in (1, 2, 3)$  do
7:         for all  $\text{ancestor} \in W.\text{ancestors}(\text{trelated}, \text{hop})$  do
8:            $\text{Rank}(\text{ancestor}) = \text{Rank}(\text{ancestor}) + 1.0/\text{hop}$ 
9:         for all  $\text{ntype} \in \text{Rank}.\text{top}(\text{top}_h)$  do
10:           $H = H \cup (\text{source}, \text{ntype})$ 
11: return  $H$ 
```

Consequently, in the last phase of our resource creation pipeline we link these ‘orphan’ PCZ senses (i.e., those from lines 17-18 of Algorithm 5), in order to obtain a unified resource, and propose a method to infer the type of those concepts that were not linked to the target lexical resource. For example, so far we were not able to find a BabelNet sense for the PCZ item **Roddenberry:10** (the author of ‘Star Trek’). However, by looking at the linked related concepts that share the same BabelNet hypernym – e.g. the PCZ items **Asimov:3** *is-a* $\text{author}_{\text{BabelNet}}$, **Tolkien:7** *is-a* $\text{author}_{\text{BabelNet}}$, **Heinlein:8** *is-a* $\text{author}_{\text{BabelNet}}$, etc. – we can infer that **Roddenberry:10** *is-a* $\text{author}:1$, since the latter was linked to the Babel synset $\text{author}_{\text{BabelNet}}$.

The input of Algorithm 6 consist of the mapping M of a PCZ to a lexical resource W (cf. Algorithm 5). The output is a new mapping H containing pairs of the kind $(\text{source}, \text{type})$ where type is a type in W for the concept $\text{source} \in \text{PCZ}$. We first initialize the new mapping H as an empty set (line 1). Then for all the pairs $(\text{source}, \text{target})$ where the target is a concept not included in the target lexical resource W (line 3), we compute a rank of all the ancestors of each related sense that has a counterpart trelated in W (lines 5-8). In other words, starting from linked related senses trelated , we traverse the taxonomy hierarchy (at most for 3 hops) in W and each time we encounter a sense ancestor we increment its rank by the inverse of the distance to trelated . Finally we add the pairs $(\text{source}, \text{ntype})$ to H for all the ntype at the top top_h in the Rank .

Finally, our final resource consists of: i) the proto-conceptualization (PCZ); ii) the mapping M of PCZ entries to the lexical resource (e.g., WordNet or BabelNet); iii) the

PCZ ID	WordNet ID	PCZ Related Terms	PCZ Context Clues
mouse:0	mouse:wn1	rat:0, rodent:0, monkey:0, ...	rat:conj_and, gray:amod, ...
mouse:1	mouse:wn4	keyboard:1, computer:0, printer:0 ...	click:-prep_of, click:-nn,
keyboard:0	keyboard:wn1	piano:1, synthesizer:2, organ:0 ...	play:-dobj, electric:amod, ..
keyboard:1	keyboard:wn1	keypad:0, mouse:1, screen:1 ...	computer, qwerty:amod ...

Table 5.1: Sample entries of the hybrid aligned resource (HAR) for the words *mouse* and *keyboard*. Trailing numbers indicate sense identifiers. To enrich WordNet sense representations we rely on related terms and context clues.

mapping H of suggested types for the PCZ entries not mapped in M .

5.3 Results

Manual evaluations against ground-truth judgments for different stages of our method as well as an extrinsic evaluation on a knowledge-based Word Sense Disambiguation benchmark all indicate the high quality of the new hybrid resource. Additionally, we show the benefits of enriching top-down lexical knowledge resources with bottom-up distributional information from text for addressing high-end knowledge acquisition tasks such as cleaning hypernym graphs and learning taxonomies from scratch.

Examples of the linked word senses are provided in Table 5.1 between Wordnet constructed manually and corpus-induced sense repository from sparse count-based distributional model (PCZ). Further examples, experimental results and their analysis can be found in [7, 8] and Appendices A.7 and A.8.

Chapter 6

Prediction of Hypernym Embeddings

Materials of this chapter are based on paper [4] from the list of 14 publications the thesis is based on. Full text is available in Appendix A.4.

6.1 Introduction

In this section, a method for extraction of hypernyms based on projection learning and word embeddings is presented. In contrast to classification-based approaches, projection-based methods require no candidate hyponym-hypernym pairs. While it is natural to use both positive and negative training examples in supervised relation extraction, the impact of negative examples on hypernym prediction was not studied so far. In this chapter, we show that explicit negative examples used for regularization of the model significantly improve performance compared to the state-of-the-art approach of [75] on three datasets from different languages.

6.2 Method

Our approach performs hypernymy extraction via regularized projection learning.

Baseline Approach In our experiments, we use the model of [75] as the baseline. In this approach, the projection matrix Φ^* is obtained similarly to the linear regression problem,

i.e., for the given row word vectors \vec{x} and \vec{y} representing correspondingly hyponym and hypernym, the square matrix Φ^* is fit on the training set of positive pairs \mathcal{P} :

$$\Phi^* = \arg \min_{\Phi} \frac{1}{|\mathcal{P}|} \sum_{(\vec{x}, \vec{y}) \in \mathcal{P}} \|\vec{x}\Phi - \vec{y}\|^2, \quad (6.1)$$

where $|\mathcal{P}|$ is the number of training examples and $\|\vec{x}\Phi - \vec{y}\|$ is the distance between a pair of row vectors $\vec{x}\Phi$ and \vec{y} . In the original method, the L^2 distance is used. To improve performance, k projection matrices Φ are learned one for each cluster of relations in the training set. One example is represented by a hyponym-hypernym offset. Clustering is performed using the k -means algorithm [76].

Linguistic Constraints via Regularization The nearest neighbors generated using distributional word vectors tend to contain a mixture of synonyms, hypernyms, co-hyponyms and other related words [77–79]. In order to explicitly provide examples of undesired relations to the model, we propose two improved versions of the baseline model: *asymmetric regularization* that uses inverted relations as negative examples, and *neighbor regularization* that uses relations of other types as negative examples. For that, we add a regularization term to the loss function:

$$\Phi^* = \arg \min_{\Phi} \frac{1}{|\mathcal{P}|} \sum_{(\vec{x}, \vec{y}) \in \mathcal{P}} \|\vec{x}\Phi - \vec{y}\|^2 + \lambda R, \quad (6.2)$$

where λ is the constant controlling the importance of the regularization term R .

Asymmetric Regularization. As hypernymy is an asymmetric relation, our first method enforces the asymmetry of the projection matrix. Applying the same transformation to the predicted hypernym vector $\vec{x}\Phi$ should not provide a vector similar (\cdot) to the initial hyponym vector \vec{x} . Note that, this regularizer requires only positive examples \mathcal{P} :

$$R = \frac{1}{|\mathcal{P}|} \sum_{(\vec{x}, \cdot) \in \mathcal{P}} (\vec{x}\Phi\Phi \cdot \vec{x})^2. \quad (6.3)$$

Neighbor Regularization. This approach relies on the negative sampling by explicitly providing the examples of semantically related words \vec{z} of the hyponym \vec{x} that penalizes

the matrix to produce the vectors similar to them:

$$R = \frac{1}{|\mathcal{N}|} \sum_{(\vec{x}, \vec{z}) \in \mathcal{N}} (\vec{x}\Phi\Phi \cdot \vec{z})^2. \quad (6.4)$$

This regularizer requires negative samples \mathcal{N} . In our experiments, we use synonyms of hyponyms as \mathcal{N} , but other types of relations can be also used such as antonyms, meronyms or co-hyponyms. Certain words might have no synonyms in the training set. In such cases, we substitute \vec{z} with \vec{x} , gracefully reducing to the previous variation. Otherwise, on each training epoch, we sample a random synonym of the given word.

Regularizers without Re-Projection. In addition to the two regularizers described above, that rely on re-projection of the hyponym vector ($\vec{x}\Phi\Phi$), we also tested two regularizers without re-projection, denoted as $\vec{x}\Phi$. The neighbor regularizer in this variation is defined as follows:

$$R = \frac{1}{|\mathcal{N}|} \sum_{(\vec{x}, \vec{z}) \in \mathcal{N}} (\vec{x}\Phi \cdot \vec{z})^2. \quad (6.5)$$

In our case, this regularizer penalizes relatedness of the predicted hypernym $\vec{x}\Phi$ to the synonym \vec{z} . The asymmetric regularizer without re-projection is defined in a similar way.

Training of the Models To learn parameters of the considered models we used the Adam method [80] with the default meta-parameters as implemented in the TensorFlow framework [81].¹ We ran 700 training epochs passing a batch of 1024 examples to the optimizer. We initialized elements of each projection matrix using the normal distribution $\mathcal{N}(0, 0.1)$.

6.3 Results

Evaluation of the proposed methods was done for one Russian and two English hypernymy datasets. Our experiments in the context of the hypernymy prediction task for both languages show significant improvements of the proposed approach over the state-of-the-art model without negative sampling.

Further experimental results and their analysis can be found in [4] and Appendix A.4.

¹<https://www.tensorflow.org>

Chapter 7

Extracting of Hypernyms via Sense Graph Clustering

Materials of this chapter are based on papers [19] from the list of 14 publications the thesis is based on. Full text is available in Appendix A.13.

7.1 Introduction

In this chapter, we show how distributionally-induced semantic classes can be helpful for extracting hypernyms. We present methods for inducing sense-aware semantic classes using distributional semantics and using these induced semantic classes for filtering noisy hypernymy relations. Denoising of hypernyms is performed by labelling each semantic class with its hypernyms. On the one hand, this allows us to filter out wrong extractions using the global structure of distributionally similar senses. On the other hand, we infer missing hypernyms via label propagation to cluster terms. We conduct a large-scale crowdsourcing study showing that processing of automatically extracted hypernyms using our approach improves the quality of the hypernymy extraction in terms of both precision and recall. Furthermore, we show the utility of our method in the domain taxonomy induction task, achieving the state-of-the-art results on a SemEval-2016 task on taxonomy induction.

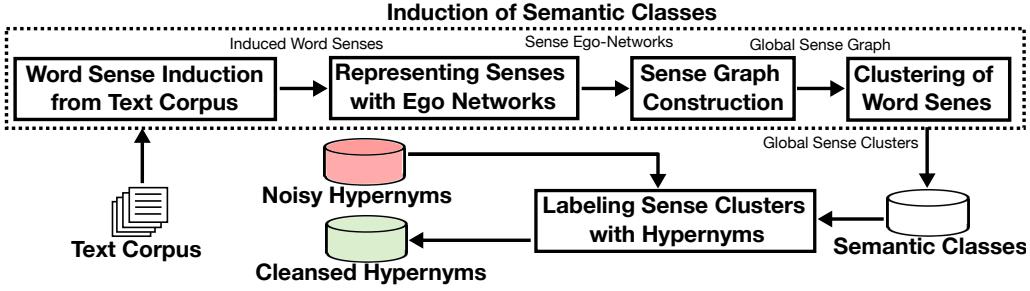


Figure 7-1: Outline of our approach: sense-aware distributional semantic classes are induced from a text corpus and then used to filter noisy hypernyms database (e.g. extracted by an external method from a text corpus).

7.2 Method

In this section, a method for unsupervised induction of distributional sense-aware semantic classes is presented. Also we show how to perform denoising of hypernyms using the induced structures. As illustrated in Figure 7-1, our method induces a sense inventory from a text corpus using the method of [82, 83], and clusters these senses.

Sample word senses from the induced semantic clusters are presented in Table 7.1. The difference of the induced sense inventory from the sense clustering is that word senses in the induced resource are specific to a given target word, e.g. words “apple” and “mango” have distinct “fruit” senses, represented by a list of related senses. On the other hand, Sense clusters represent a global and not a local clustering of senses, i.e. the “apple” in the “fruit” sense can be a member of only one cluster. This is similar to WordNet, where one sense can only belong to a single synset. Below we describe each step of our method.

Word Sense Induction from a Text Corpus

Each word sense s in the induced sense inventory \mathcal{S} is represented by a list of neighbors $\mathcal{N}(s)$. Extraction of this network is performed using the method of [82] and involves three steps: (1) building a distributional thesaurus, i.e. a graph of related ambiguous terms [84]; (2) word sense induction via clustering of ego networks [85, 86] of related words using the Chinese Whispers graph clustering algorithm [46]; (3) disambiguation of related words and hypernyms. The word sense inventory used in our experiment was extracted from a 9.3 billion tokens corpus, which is a concatenation of Wikipedia , ukWac [87], LCC [88] and Gigaword [89]. Note that analogous graphs of senses can be obtained using word sense embeddings, see [65, 90]. Similarly to any other distributional word graph, the induced

Global Sense Cluster: Semantic Class, $c \subset S$	Hypernyms, $\mathcal{H}(c) \subset S$
peach#1, banana#1, pineapple#0, berry#0, blackberry#0, grape-fruit#0, strawberry#0, blueberry#0, fruit#0, grape#0, melon#0, orange#0, pear#0, plum#0, raspberry#0, watermelon#0, apple#0, apricot#0, watermelon#0, pumpkin#0, berry#0, mangosteen#0 , ...	vegetable#0, fruit#0, crop#0, ingredient#0, food#0, .
C#4, Basic#2, Haskell#5, Flash#1, Java#1, Pascal#0, Ruby#6, PHP#0, Ada#1, Oracle#3, Python#3, Apache#3, Visual Basic#1, ASP#2, Delphi#2, SQL Server#0, CSS#0, AJAX#0, JavaScript#0, SQL Server#0, Apache#3, Delphi#2, Haskell#5, .NET#1, CSS#0, ...	programming language#3, technology#0, language#0, format#2, app#0

Table 7.1: Sample of the induced sense clusters representing “fruits” and “programming language” semantic classes. Similarly to the induced word senses, the semantic classes are labeled with hypernyms. In contrast to the induced word senses, which represent a local clustering of word senses (related to a given word) semantic classes represent a global sense clustering of word senses.

sense inventory sense network is scale-free, cf. [91]. Our experiments show that a global clustering of this network can lead to a discovery of giant components, which are useless in our context as they represent no semantic class. To overcome this problem, we re-build the sense network as described below.

Representing Senses with Ego Networks

To perform a global clustering of senses, we represent each induced sense s by a second-order *ego network* [86]. An ego network is a graph consisting of all related senses $\mathcal{R}(s)$ of the ego sense s reachable via a path of length one or two, defined as:

$$\{s_j : (s_j \in \mathcal{N}(s)) \vee (s_i \in \mathcal{N}(s) \wedge s_j \in \mathcal{N}(s_i))\}. \quad (7.1)$$

Each edge weight $\mathcal{W}_s(s_i, s_j)$ between two senses is taken from the induced sense inventory network [82] and is equal to a distributional semantic relatedness score between s_i and s_j .

Senses in the induced sense inventory may contain a mixture of different senses introducing noise in a global clustering, e.g. “Python” in the animal sense is related to both car and snake senses. To minimize the impact of the word sense induction errors, we filter out ego networks with a highly segmented structure. Namely, we cluster each ego network with the Chinese Whispers algorithm and discard networks for which the cluster containing the target sense s contains less than 80% nodes of the respective network to ensure semantic coherence inside the word groups. Besides, all nodes of a network not

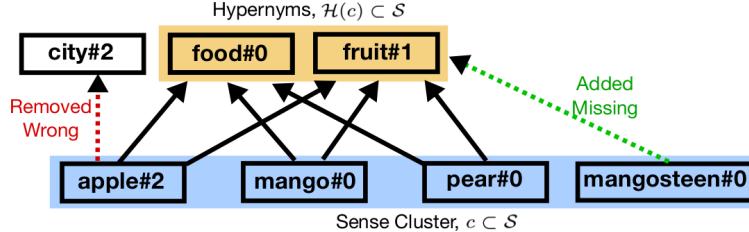


Figure 7-2: Our approach performs post-processing of hypernymy relations using distributionally induced semantic classes, represented by clusters of induced word senses labeled with noisy hypernyms. The word postfix, such as #1, is an ID of an induced sense. The wrong hypernyms outside the cluster labels are removed, while the missing ones not present in the noisy database of hypernyms are added.

appearing in the cluster containing the ego sense s are also discarded.

Global Sense Graph Construction

The goal of this step is to merge ego networks of individual senses constructed at the previous step into a global graph. We compute weights of the edges of the global graph by counting the number of co-occurrences of the same edge in different networks:

$$\mathcal{W}(s_i, s_j) = \sum_{s \in \mathcal{S}} \mathcal{W}_s(s_i, s_j). \quad (7.2)$$

For filtering out noisy edges, we remove all edges with the weight less than a threshold t . Finally, we apply the function $E(w)$ that re-scales edge weights. We tested identity function (count) and the natural logarithm (log):

$$\mathcal{W}(s_i, s_j) = \begin{cases} E(\mathcal{W}(s_i, s_j)) & \text{if } \mathcal{W}(s_i, s_j) \geq t, \\ 0 & \text{otherwise.} \end{cases} \quad (7.3)$$

Clustering of Word Senses

The core of our method is the induction of semantic classes by clustering the global graph of word senses. We use the Chinese Whispers algorithm to make every sense appear only in one cluster c . Results of the algorithm are groups of strongly related word senses that represent different concepts. Hypernymy is by definition a relation between nouns.

We use two clustering versions in our experiments: the *fine-grained* model clusters 208,871 induced word senses into 1,870 semantic classes, and the *coarse-grained* model that groups 18,028 word senses into 734 semantic classes. To find optimal parameters of our

method, we compare the induced labeled sense clusters to lexical semantic knowledge from WordNet 3.1 [92] and BabelNet 3.7 [93].

Denoising Hypernyms using the Induced Distributional Semantic Classes

By labeling the induced semantic classes with hypernyms we can thereby remove wrong ones or add those that are missing as illustrated in Figure 7-2. Each sense cluster is labeled with the noisy input hypernyms, where the labels are the common hypernyms of the cluster word (cf. Table 7.1). Hypernyms that label no sense cluster are filtered out. In addition, new hypernyms can be generated as a result of labeling. Additional hypernyms are discovered by propagating cluster labels to the rare words without hypernyms, e.g. “mangosteen” in Figure 7-2. For labeling we used the tf-idf weighting. Hypernyms that appear in many senses s are weighted down:

$$\text{tf-idf}(h) = \sum_{s \in c} \mathcal{H}(s) \cdot \log \frac{|\mathcal{S}|}{|h \in \mathcal{H}(s) : \forall s \in \mathcal{S}|}, \quad (7.4)$$

where $\sum_{s \in c} \mathcal{H}(s)$ is a sum of weights for all hypernyms for each sense s , per each cluster c . We label each sense cluster c with its top five hypernyms $\mathcal{H}(c)$. Each hypernym is disambiguated using the method of [82]. Namely, we calculate the cosine similarity between the context (the current sense cluster) and the induced senses (local clusters of the ambiguous word).

Distributional representations of rare words, such as “mangosteen” can be less precise than those of frequent words. However, co-occurrence of a hyponym and a hypernym in a single sentence is not required in our approach, while it is the case for the path-based hypernymy extraction methods.

7.3 Results

To evaluate approach three experiments were conducted. A large-scale crowdsourcing study indicated a high plausibility of extracted semantic classes according to human judgment. Besides, we demonstrated that our approach helps to improve precision and recall of a hypernymy extraction method. Finally, we showed how the proposed semantic classes can be used to improve domain taxonomy induction from text (based on SemEval-2016 dataset).

Further experimental results and their analysis can be found in [19] and Appendix A.13.

Chapter 8

Taxonomy Enrichment using Hyperbolic Embeddings

Materials of this chapter are based on papers [5] from the list of 14 publications the thesis is based on. Full text is available in Appendix A.5.

8.1 Introduction

The task of taxonomy enrichment can be formulated as following. Given a noisy taxonomic graph $G = (V, E)$ with errors of two types (i) *Absent edges*: $E_{abs} = \{(v_i, v_j) \in E : v_i \text{ is-a } v_j \wedge (v_i, v_j) \notin E\}$ with a special case of *orphan nodes* $V_{orh} = \{v_i \in V : \nexists (v_i, v_j) \in E\}$; and (ii) *Wrong edges*: $E_{wrg} = \{(v_i, v_j) \in E : v_i \text{ not-is-a } v_j\}$ build a graph $G' = (V, E')$ correcting the two edge errors by: (i) *adding absent edges*: $E = E \cup \{E_{abs}\}$; (ii) *removing wrong edges*: $E = E \setminus E_{wrg}$. For orphan nodes only adding edges is needed. For connected nodes either adding absent additional edge is needed or *relocation* i.e. a combination of removing wrong with adding absent edge(s) is needed.

We introduce the use of Poincaré embeddings to improve existing state-of-the-art approaches to domain-specific taxonomy induction from text as a signal for both relocating wrong hyponym terms within a (pre-induced) taxonomy as well as for attaching disconnected terms in a taxonomy. This method substantially improves previous state-of-the-art results on taxonomy extraction. We demonstrate the superiority of Poincaré embeddings over distributional semantic representations, supporting the

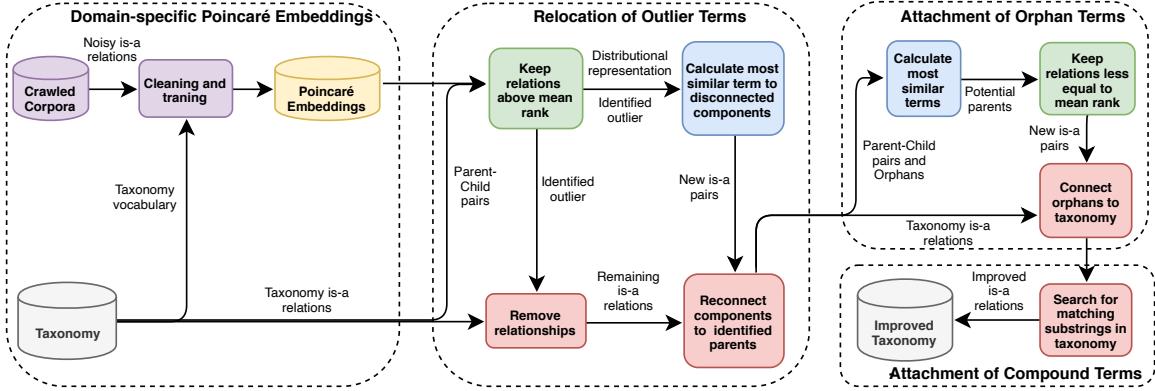


Figure 8-1: Outline of our taxonomy refinement method.

hypothesis that they can better capture hierarchical lexical-semantic relationships than embeddings in the Euclidean space.

8.2 Method

In this chapter, a method for taxonomy refinement using hyperbolic word embedding is presented. We employ embeddings using distributional semantics (i.e. word2vec) and Poincaré embeddings [94] to alleviate the largest error classes in taxonomy extraction: the existence of *orphans* – disconnected nodes that have an overall connectivity degree of zero and *outliers* – a child node that is assigned to a wrong parent. The rare case in which multiple parents can be assigned to a node has been ignored in the proposed refinement system. The first step consists of creating domain-specific Poincaré embeddings. They are then used to identify and relocate outlier terms in the taxonomy, as well as to attach unconnected terms to the taxonomy. In the last step, we further optimize the taxonomy by employing the endocentric nature of hyponyms. See Figure 8-1 for a schematic visualization of the refinement pipeline.

Training Dataset Construction

To create domain-specific Poincaré embeddings, we use noisy hypernym relationships extracted from a combination of general and domain-specific corpora. For the general domain, we extracted text from English Wikipedia, Gigaword [95], ukWac [96] and LCC news corpora [97]. Noisy IS-A relations are extracted with lexical-syntactic patterns from all corpora by applying PattaMaika,PatternSim [98], and WebISA [99] following [100].

The extracted noisy relationships of the common and domain-specific corpora are

further processed separately and combined afterward. To limit the number of terms and relationships, we restrict the IS-A relationships on pairs for which both entities are part of the taxonomy’s vocabulary. Relations with a frequency of less than three are removed to filter noise. Besides further removing every reflexive relationship, only the more frequent pair of a symmetric relationship is kept. Hence, the set of cleaned relationships is transformed into being antisymmetric and irreflexive.

The same procedure is applied to relationships extracted from the general-domain corpus. They are then used to expand the set of relationships created from the domain-specific corpora.

Hypernym-Hyponym Distance

Poincaré embeddings are trained on these cleaned IS-A relationships. For comparison, we also trained a model on noun pairs extracted from WordNet (P-WN). Pairs were only kept if both nouns were present in the vocabulary of the taxonomy. Finally, we trained the word2vec embeddings, connecting compound terms in the training corpus (Wikipedia) by ‘_’ to learn representations for compound terms, i.e multiword units, for the input vocabulary.

In contrast to embeddings in the Euclidean space where the cosine similarity is commonly applied as a similarity measure, i.e.

$$d(\mathbf{u}, \mathbf{v}) = 1 - \frac{\mathbf{u} \cdot \mathbf{v}}{|\mathbf{u}| |\mathbf{v}|}, \quad (8.1)$$

Poincaré embeddings use a hyperbolic space, specifically the Poincaré ball model [101]. Hyperbolic embeddings are designed for modeling hierarchical relationships between words as they explicitly capture the hierarchy between words in the embedding space and are therefore a natural fit for inducing taxonomies. They were also successfully applied to hierarchical relations in image classification tasks [102]. The distance between two points $\mathbf{u}, \mathbf{v} \in \mathcal{B}^d$ for a d -dimensional Poincaré Ball model is defined as:

$$d(\mathbf{u}, \mathbf{v}) = \text{arcosh}\left(1 + 2 \frac{||\mathbf{u} - \mathbf{v}||^2}{(1 - ||\mathbf{u}||^2)(1 - ||\mathbf{v}||^2)}\right). \quad (8.2)$$

This Poincaré distance enables us to capture the hierarchy and similarity between words simultaneously. It increases exponentially with the depth of the hierarchy. So while the distance of a leaf node to most other nodes in the hierarchy is very high, nodes on abstract levels, such as the root, have a comparably small distance to all nodes in the hierarchy. The

word2vec embeddings have no notion of hierarchy and hierarchical relationships cannot be represented with vector offsets across the vocabulary [103]. When applying word2vec, we use the observation that distributionally similar words are often co-hyponyms [104, 105].

Relocation of Outlier Terms

Poincaré embeddings are used to compute and store a rank $rank(x, y)$ between every child and parent of the existing taxonomy, defined as the index of y in the list of sorted Poincaré distances of all entities of the taxonomy to x . Hypernym-hyponym relationships with a rank larger than the mean of all ranks are removed, chosen on the basis of tests on the 2015 TExEval data [106]. Disconnected components that have children are re-connected to the most similar parent in the taxonomy or to the taxonomy root. Previously or now disconnected isolated nodes are subject to orphan attachment.

Since distributional similarity does not capture parent-child relations, the relationships are not registered as parent-child but as co-hyponym relationships. Thus, we compute the distance to the closest co-hyponym (child of the same parent) for every node. This filtering technique is then applied to identify and relocate outliers.

Attachment of Orphan Terms

We then attach orphans (nodes unattached in the input or due to the removal of relationships in the previous step) by computing the rank between every orphan and the most similar node in the taxonomy. This node is an orphan’s potential parent. Only hypernym-hyponym relationships with a rank lower or equal to the mean of all stored ranks are added to the taxonomy. For the word2vec system, a link is added between the parent of the most similar co-hyponym and the orphan.

Attachment of Compound Terms

In case a representation for a compound noun term does not exist, we connect it to a term that is a substring of the compound. If no such term exists, the noun remains disconnected. Finally, the Tarjan algorithm [107] is applied to ensure that the refined taxonomy is asymmetric: In case a circle is detected, one of its links is removed at random.

8.3 Results

Evaluation of the proposed method was performed on SemEval-2016 taxonomy enrichment dataset and output of three top systems. The refinement method is generically applicable

Word	Parent patterns	Parent after refinement	Gold parent	Closest neighbors
second language acquisition	—	linguistics	linguistics	applied linguistics, semantics, linguistics
botany	—	genetics	plant science, ecology	genetics, evolutionary ecology, animal science
sweet potatoes	—	vegetables	vegetables	vegetables, side dishes, fruit
wastewater	water	waste	waste	marine pollution, waste, pollutant
water	waste, natural resources	natural resources	aquatic environment	continental shelf, management of resources
international relations	sociology, analysis, humanities	humanities	political science	economics, economic theory, geography

Table 8.1: Example words with respective parent(s) in the input taxonomy constructed using Hearst’ patterns approach and after refinement using our domain-specific Poincaré embeddings, as well as the word’s closest three neighbors (incl. orphans) in embeddings.

to noisy taxonomies, yielding an improved taxonomy extraction system overall. Examples of predictions are available in Table 8.1. Experiments show that, the developed refinement method for improving existing taxonomies through the use of hyperbolic Poincaré embeddings consistently yield improvements over strong baselines and in comparison to word2vec as a representative for distributional vectors in the Euclidean space. We further showed that Poincaré embeddings can be efficiently created for a specific domain from crawled text without the need for an existing database such as WordNet. This observation confirms the theoretical capability of Poincaré embeddings to learn hierarchical relations, which enables their future use in a wide range of semantic tasks.

Further experimental results and their analysis can be found in [5] and Appendix A.5.

Chapter 9

Node Embeddings of Lexical-Semantic Graphs

Materials of this chapter are based on paper [3] from the list of 14 publications the thesis is based on. Full text is available in Appendix A.3.

9.1 Introduction

When operating on large graphs, such as transportation networks, social networks, or lexical resources, the need for estimating similarities between nodes arises. For many domain-specific applications, custom graph node similarity measures $sim : V \times V \rightarrow \mathbb{R}$ have been defined on pairs of nodes V of a graph $G = (V, E)$. Examples include travel time, communities, or semantic distances for knowledge-based word sense disambiguation on WordNet [108]. For instance, the similarity s_{ij} between the `cup.n.01` and `mug.n.01` synsets in the WordNet is $\frac{1}{4}$ according to the inverted shortest path distance as these two nodes are connected by the undirected path `cup → container ← vessel ← drinking_vessel ← mug`.

In recent years, a large variety of such node similarity measures have been described, many of which are based on the notion of a random walk [109–111]. As given by the structure of the problem, most such measures are defined as traversals of edges E of the graph, which makes their computation prohibitively inefficient.

To this end, we propose the *path2vec* model, which solves this problem by decoupling

development and use of graph-based measures, and – in contrast to purely walk- based embeddings – is trainable to reflect custom node similarity measures. We represent nodes in a graph with dense embeddings that are good in approximating such custom, e.g. application-specific, pairwise node similarity measures. Similarity computations in a vector space are several orders of magnitude faster than computations directly operating on the graph.

9.2 Method

Definition of the Model

Path2vec is a graph metric embeddings model learns embeddings of the graph nodes $\{v_i, v_j\} \in V$ such that the dot products between pairs of the respective vectors ($\mathbf{v}_i \cdot \mathbf{v}_j$) are close to the user-defined similarities between the nodes s_{ij} . In addition, the model reinforces the similarities $\mathbf{v}_i \cdot \mathbf{v}_n$ and $\mathbf{v}_j \cdot \mathbf{v}_m$ between the nodes v_i and v_j and all their respective adjacent nodes $\{v_n : \exists(v_i, v_n) \in E\}$ and $\{v_m : \exists(v_j, v_m) \in E\}$ to preserve local structure of the graph. The model preserves both **global** and **local** relations between nodes by minimizing

$$\mathcal{L} = \sum_{(v_i, v_j) \in B} ((\mathbf{v}_i^\top \mathbf{v}_j - s_{ij})^2 - \alpha(\mathbf{v}_i^\top \mathbf{v}_n + \mathbf{v}_j^\top \mathbf{v}_m)), \quad (9.1)$$

where $s_{ij} = sim(v_i, v_j)$ is the value of a ‘gold’ similarity measure between a pair of nodes v_i and v_j , \mathbf{v}_i and \mathbf{v}_j are the embeddings of the first and the second node, B is a training batch, α is a regularization coefficient. The second term ($\mathbf{v}_i \cdot \mathbf{v}_n + \mathbf{v}_j \cdot \mathbf{v}_m$) in the objective function is a regularizer that aids the model to simultaneously maximize the similarity between adjacent nodes while learning the similarity between the two target nodes (one adjacent node is randomly sampled for each target node).

We use negative sampling to form a training batch B adding p negative samples ($s_{ij} = 0$) for each real ($s_{ij} > 0$) training instance: each real node (synset) pair (v_i, v_j) with ‘gold’ similarity s_{ij} is accompanied with p ‘negative’ node pairs (v_i, v_k) and (v_j, v_l) with zero similarities, where v_k and v_l are randomly sampled nodes from V . Embeddings are initialized randomly and trained using the *Adam* optimizer [112] with early stopping. Once the model is trained, the computation of node similarities is approximated with the dot product of the learned node vectors, making the computations efficient: $\hat{s}_{ij} = \mathbf{v}_i \cdot \mathbf{v}_j$.

Relation to Similar Models

Our model bears resemblance to the Skip-gram model [113], where the vector dot product $\mathbf{v}_i \cdot \tilde{\mathbf{v}}_j$ of vectors of pairs of words (v_i, v_j) from a training corpus is optimized to a high score close to 1 for observed samples, while the dot products of negative samples are optimized towards 0. In the Skip-gram model, the target is to minimize the log likelihood of the conditional probabilities of context words w_j given current words w_i :

$$\mathcal{L} = - \sum_{(v_i, v_j) \in B_p} \log \sigma(\mathbf{v}_i \cdot \tilde{\mathbf{v}}_j) - \sum_{(v_i, v_j) \in B_n} \log \sigma(-\mathbf{v}_i \cdot \tilde{\mathbf{v}}_j), \quad (9.2)$$

where B_p is the batch of positive training samples, B_n is the batch of the generated negative samples, and σ is the sigmoid function. At this, Skip-gram uses only [local](#) information, never creating the full co-occurrence count matrix. In our *path2vec* model, the target dot product values s_{ij} are not binary, but can take arbitrary values in the $[0...1]$ range, as given by the custom distance metric. Further, we use only a single embedding matrix with vector representations of the graph nodes, not needing to distinguish target and context.

Another related model is Global Vectors (GloVe) [114], which learns co-occurrence probabilities in a given corpus. The objective function to be minimized in GloVe model is $\mathcal{L} = \sum_{(v_i, v_j) \in B} f(s_{ij})(\mathbf{v}_i \cdot \tilde{\mathbf{v}}_j - \log s_{ij} + b_i + b_j)^2$, where s_{ij} counts the co-occurrences of words v_i and v_j , b_i and b_j are additional biases for each word, and $f(s_{ij})$ is a weighting function handling rare co-occurrences. Like the Skip-gram, GloVe also uses two embedding matrices, but it relies only on [global](#) information, pre-aggregating global word co-occurrence counts.

Computing Training Similarities

In general case, our model requires computing pairwise node similarities s_{ij} for training between all pairs of nodes in the input graph G . This step could be computationally expensive, but it is done only once to make computing of similarities fast. Besides, for some metrics, effective algorithms exist that compute all pairwise similarities at once, e.g. [115] algorithm for computing shortest paths distances with the worst-case performance of $O(|V|^2 \log |V| + |V||E|)$. As the input training dataset also grows quadratically in $|V|$, training time for large graphs can be slow. To address this issue, we found it useful to prune the input training set so that each node $v_i \in V$ has only $k \in [50; 200]$ most similar nodes. Such pruning does not lead to loss of effectiveness.

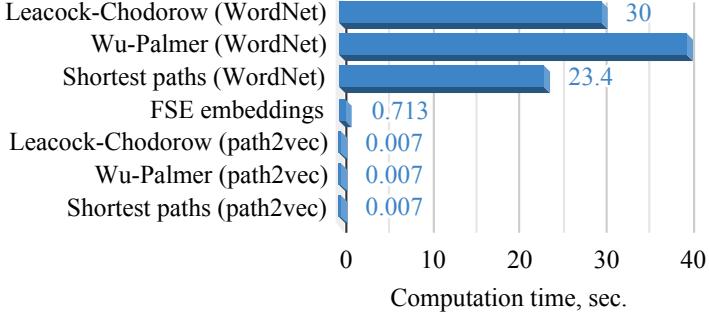


Figure 9-1: Similarity computation: graph vs vectors.

9.3 Results

Experiments were conducted to measure computational efficiency of the algorithm, as well as quality of approximation intrinsically (based on SimLex999 dataset) and extrinsically based on three SemEval WSD datasets. A comparison to baseline approaches, such as Deepwalk, node2vec or TransR were conducted. We demonstrated that our approach generalizes well across graphs (WordNet, Freebase, and DBpedia). Besides, we integrated it into a graph-based WSD algorithm, showing that its vectorized counterpart yields comparable F1 scores for the WSD task.

Path2vec enables a speed-up of up to four orders of magnitude for the computation of graph distances as compared to ‘direct’ graph measures (see Figure 9-1). Thus, our model is simple and general, hence it may be applied to any graph together with a node distance measure to speed up algorithms that employ graph distances.

Structured knowledge contained in language networks is useful for NLP applications but is difficult to use directly in neural architectures. We proposed a way to train embeddings that directly represent a graph-based similarity measure structure. Our model, *path2vec*, relies on both global and local information from the graph and is simple, effective, and computationally efficient.

Further experimental results and their analysis can be found in [3] and Appendix A.3.

Chapter 10

Lexical Substitution and Analysis of its Semantic Relation Types

Materials of this chapter are based on paper [2] from the list of 14 publications the thesis is based on. Full text is available in Appendix A.2.

10.1 Introduction

Lexical substitution, i.e. generation of plausible words that can replace a particular target word in a given context, is a powerful technology that can be used as a backbone of various NLP applications, such as word sense induction [116], lexical relation extraction [117], paraphrase generation, text simplification, textual data augmentation, etc. Note that the preferable type (e.g., synonym, hypernym, co-hyponym, etc.) of generated substitutes depends on the task at hand.

We present a study of lexical substitution methods employing both classic and more recent language and masked language models (LMs and MLMs), such as context2vec, ELMo, BERT, RoBERTa, XLNet. We show that already competitive results achieved by SOTA LMs/MLMs can be further substantially improved if information about the target word is injected properly. Besides, we perform analysis of lexical semantic relation types generated by these models, as illustrated in Figure 10-1.

We were not able to travel in the weather , and there was no phone .											
GOLD	telephone (5)	telephone	phones	cellphone	fone	videophone	handset	telephones	p990i	cell-phone	
OOC	phone	telephone	phones	cellphone	fone	videophone	handset	telephones	p990i	cell-phone	
XLNet	electricity	internet	phone	power	telephone	car	water	communication	radio	tv	
XLNet+embs	phone	telephone	phones	cellphone	internet	radio	electricity	iphone	car	computer	
What happened to the big , new garbage can at Church and Chambers Streets ?											
GOLD	bin (4)	disposal (1)	container (1)								
OOC	can	could	should	would	will	must	might	to	may	ll	
XLNet	can	dump	bin	truck	disposal	pit	heap	pile	container	stand	
XLNet+embs	can	could	will	bin	cannot	dump	may	truck	disposal	stand	

Types of semantic relations: synonym co-hyponym co-hyponym 3 target direct hypernym transitive hypernym
 direct hyponym transitive hyponym unknown-relation unknown-word

Figure 10-1: Examples of top substitutes provided by annotators (GOLD), the baseline (OOC), and two presented models (XLNet and XLNet+embs). The target word in each sentence is in bold, true positives are in bold also. The weights of gold substitutes are given in brackets. Each substitute is colored according to its lexical-semantic relation to the target word.

10.2 Method

Lexical substitution is the task of generating words that can replace a given word in a given textual context. For instance, in the sentence “*My daughter purchased a new car*” the word *car* can be substituted by its synonym *automobile*, but also with co-hyponym *bike*, or even hypernym *motor vehicle* while keeping the original sentence grammatical.

To generate substitutes, we introduce several substitute probability estimators, which are models taking a text fragment and a target word position in it as input and producing a list of substitutes with their probabilities. To build our substitute probability estimators we employ the following LMs/MLMs: context2vec [118], ELMo [119], BERT [120], RoBERTa [121] and XLNet [122]. These models were selected to represent the progress in unsupervised pre-training with language modeling and similar tasks.

Given a target word occurrence, the basic approach for models like context2vec and ELMo is to encode its context and predict the probability distribution over possible center words in this particular context. This way, the model does not see the target word. For MLMs, the same result can be achieved by masking the target word. This basic approach employs the core ability of LMs/MLMs of predicting words that fit a particular context. However, these words are often not related to the target. The information about the target word can improve generated substitutes, but what is the best method of injecting this information is an open question.

We describe method to introduce information about the original target word into neural lexical substitution models. Suppose we have an example LTR , where T is the target word, and $C = (L, R)$ is its context (left and right, correspondingly). For instance, the occurrence of the target word *fly* in the sentence “*Let me fly away!*” will be represented as $T = “fly”$, $L = “Let me”$, $R = “away!”$.

The proposed method combines a distribution provided by a context-based substitute probability estimator $P(s|C)$ with a distribution based on the proximity of possible substitutes to the target $P(s|T)$. The proximity is computed as the inner product between the respective embeddings, and the softmax function is applied to get a probability distribution. However, if we simply multiply these distributions, the second will have almost no effect because the first is very peaky. To align the orders of distributions, we use temperature softmax with carefully selected temperature hyperparameter:

$$P(s|T) \propto \exp\left(\frac{\langle emb_s, emb_T \rangle}{\tau}\right). \quad (10.1)$$

Target word injection methods rely on word embeddings similarity and is denoted as “+embs”. The final distribution is obtained by the formula

$$P(s|C, T) \propto \frac{P(s|C)P(s|T)}{P(s)^\beta}. \quad (10.2)$$

For $\beta = 1$, this formula can be derived by applying the Bayes rule and assuming conditional independence of C and T given s . Other values of β can be used to penalize frequent words, more or less. Our current methods are limited to generating only substitutes from the vocabulary of the underlying LM/MLM. Thus, we take word or subword embeddings of the same model we apply the injection to.

Word probabilities $P(s)$ are retrieved from word frequencies for all models except ELMo. Following [123], for ELMo, we calculate word probabilities from word ranks in the ELMo vocabulary (which is ordered by word frequencies) based on Zipf-Mandelbrot distribution.

In addition to this method simpler injection strategies were tested e.g. dynamic patterns, duplicate input or original input. These showed weaker results. Different LMs/MLMs are employed as described below to obtain context-based substitute probability distribution $P(s|C)$. For each of them, we experiment with different target injection methods: context2vec, ELMo, BERT/RoBERTa, XLNet.

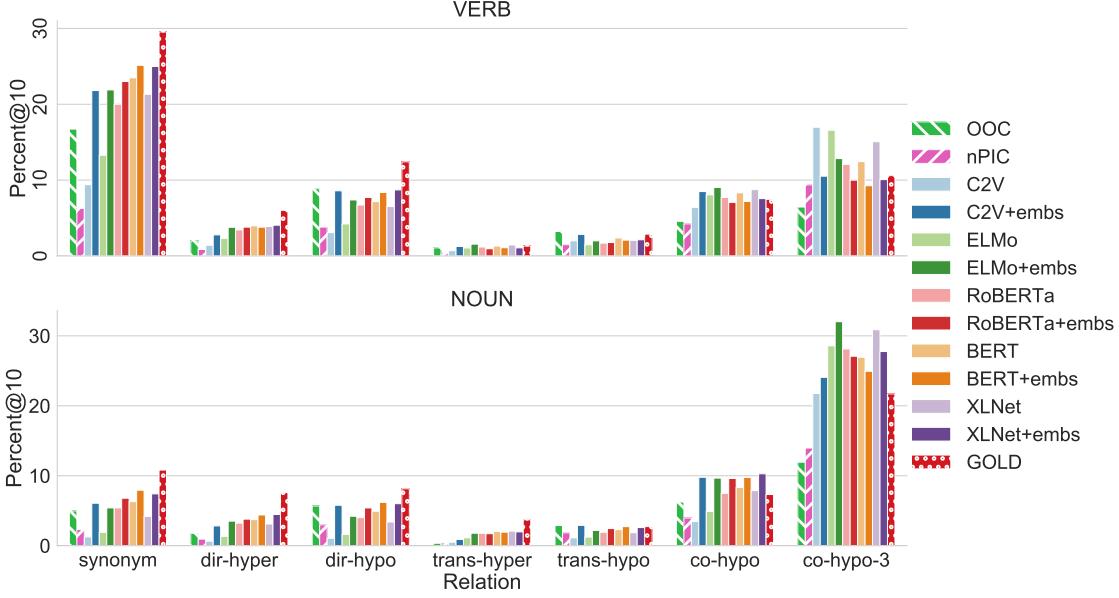


Figure 10-2: Proportions of substitutes related to the target by various semantic relations according to WordNet. We took top 10 substitutes from each model and all substitutes from the gold standard.

10.3 Results

Several existing and new target word injection methods are compared for each LM/MLM using both intrinsic evaluation on lexical substitution datasets (SemEval-2007 and CoInCo) and extrinsic evaluation on word sense induction (WSI) datasets (SemEval-2010 and SemEval-2013). Experimental results on two WSI datasets show that the proposed lexical substitution method with injection of information about target word (+embs) consistently outperforms baseline methods, such as C2V, ELMo, BERT, RoBERTa, XLNet, without such information and ranks favourably to the previous SOTA results.

Besides, we analyse the types of semantic relations, such as synonymy, hypernymy, and co-hyponymy, between target words and their substitutes generated by different models or given by annotators as illustrated in Figure 10-2. Results suggest that, co-hyponyms dominate among substitutes for nouns while synonyms dominate for verbs with co-hyponyms representing second largest type.

Further experimental results and their analysis can be found in [2] and Appendix A.2.

Chapter 11

Conclusion

The results of this thesis are based on work presented in a series of research papers and journal articles at various international venues on methods and algorithms of computational lexical semantics with application to extraction of word senses, hypernymy relations between words, and semantic frames [1–42]. More specifically points defended in this dissertation are based on 14 publications [1–10, 17–20].

The set of results touched in the mentioned publication provide a comprehensive computational framework for processing of meaning of words and phrases and relations between them (the core topic of lexical semantics). Two types of computer-readable representations of lexical units (e.g., words and phrases) and semantic relations between them are currently established: (i) resources constructed manually, such as WordNet, BabelNet, or FrameNet and, (ii) automatically induced lexical semantic representation from raw text, such as sparse or dense word embeddings, and relations extracted with rules or statistical taggers. These two representations of lexical units and relations between them were traditionally separated. Some NLP methods would rely solely on precise yet limited in recall manually constructed lexical resources, while others would only on distributional methods which are more prone to inconsistencies of semantic representation yet provide higher recall as they are obtained from huge unlabelled text corpora. In this dissertation we made several methodological contributions related to both representations and their combination.

All in all, contributions presented in this thesis can be grouped as following:

- **Graph clustering algorithm for linguistic networks:** A fuzzy meta algorithm

for graph clustering for the needs of large linguistic graph processing. The algorithm is tested successfully to extract synsets, semantic frames, and semantic classes.

- **Word sense embeddings:** Methods for learning sense embeddings using graph clustering and a mechanism for disambiguation of words with respect to these representations.
- **Inducing interpretable word sense representations:** Methods for inducing using graph-based methods word sense representations and making them interpretable by automatic retrieval of hypernyms, images, and definitions therefore making induced representations of word senses similar to those constructed manually.
- **Alignment of word sense representations:** Technology for alignment of manually created and automatically induced (such as those mentioned above) lexical representations. This set of methods allows to get 'best of both worlds' through enrichment of precise yet low coverage manual representations with high coverage word senses induced from text.
- **Disambiguation of word senses in context:** Methods for disambiguation of word meaning in context were proposed in form of classic WSD (word sense disambiguation) setup where given a word-context pair a sense identifier has to be predicted, and in the form of LexSub (lexical substitution setup) where given word-context pair a synonym fit to meaning of the context has to be predicted.
- **Induction of semantic trees:** Methods for dealing with special kind of lexical semantic resources composed of hierarchical relations, namely taxonomies, were proposed. Methods for population of existing, e.g. manually constructed, semantic trees with new nodes, e.g. new words and phrases not covered by vocabulary of such taxonomy. Besides, an approach based on clustering of graphs of automatically induced word senses was devised to perform cleansing of automatically extracted hypernymy relations.
- **Vectorisation of lexical semantic networks:** Methods for vectorisation of linguistic resources, such as lexical databases and knowledge graphs. These yield

node embeddings of linguistic networks which were applied on for completion of missing edges (resource construction) and word sense disambiguation. Besides, these vector representations may be potentially useful for other lexical-semantic tasks.

Therefore, our methodological contributions cover both aspects of automatic construction of distributional and symbolic representations of word senses. Besides, methods for linking these representation and disambiguation in context are presented. The methodology is heavily based on graph-based methods, notably graph clustering, showing that its a versatile tool in various task of computational lexical semantics. Most developed methods rely on graph representations as lexical semantic resources are naturally represented as graphs (with nodes being word senses and semantic relations being edges). At the same time, vector representations are also used demonstrating the usefulness of duality of graph-vector representation metaphor for various tasks of computational lexical semantics.

A promising direction of the future work is to investigate more large language models (LLMs), such as T5, GPT, or LLaMa for the task of generation and completion of lexical semantic resources and other tasks related to modelling meaning of words senses and relations between them.

Acknowledgements

First of all, I would like to thank Chris Biemann as the main advisor and supporter in the scientific journey covered in publications assembled in this dissertation. Secondly, I would like to thank all my co-authors. Here I would like to specifically highlight and express my gratitude to dear collaborators without whom this work would not be possible and played key role in shaping contributions presented in this thesis: Dmitry Ustalov, Nikolay Arefyev, Simono Paolo Ponzetto, Stefano Faralli, and Andrei Kutuzov. In preparation of this work part-of co-authors were students including Bachelor, Master, and PhD level to whose work and contribution I also highly appreciate and acknowledge: Ramy Aly, Maria Pelevina, Shantanu Acharya, Eugen Ruppert, Boris Sheludko, Mohamman Dorgham, Oleksiy Oliynyk, Alexander Ossa, Arne Kohn, Yuri Arkhipov, Saba Anwar, and Özge Sevgili. Its only though this extensive collaboration with both

established researchers and students of all levels this work was made possible and fun to do.

Besides, I would like to thank Yuri Nikolaevich Philippovich and Cédrick Fairon for introducing me into the field of Natural Language Processing: while from different angles and perspectives both being complementary and certainly shaping me as a curious independent researcher and life-long learner. In fact, this thesis is a logical continuation of research started during my Engineer’s graduation diploma on “Automatic thesaurus construction” at Bauman Moscow State Technical University in 2008 and PhD thesis in Université catholique de Louvain on “Similarity measures for semantic relation extraction” [51] in 2013.¹. In between first initiation to the domain and this dissertation lies over 15 years of my initiation to NLP research and development (and the topic of computational lexical semantics more specifically), spanning study, work, and life in France, Belgium, Germany and Russia not taking into account uncountable travels to conferences, scientific presentations, and encounters.

Also, I would like to thank Elena Gryazina for consultations on preparing this manuscript and Elena Tutubalina for inspiration to write.

It is paramount to acknowledge financial support from various sources and institutions which made this research possible and provided me an ample amount of free time for research and writing. The research conducted in this thesis was with support of DFG-funded project “Joining semantics of ontologies and text” (JOIN-T and JOIN-T 2) with colleagues and students from Germany, but also by a Skoltech Research Package and RSF grant “Cross-lingual Knowledge Base Construction and Completion” allowed to continue this line of work on related topics with colleagues and students from Russia.

Last but not least, I would like to thank strong support of my family: Luidmila Borisovna, Ivan Ivanovich, Polina, Evgenii, and Konstantin.

¹<https://dial.uclouvain.be/pr/boreal/object/boreal:125438>

Bibliography

- [1] A. Panchenko, E. Ruppert, S. Faralli, S. P. Ponzetto, and C. Biemann, “**Unsupervised does not mean uninterpretable: The case for word sense induction and disambiguation**,” in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, (Valencia, Spain), pp. 86–98, Association for Computational Linguistics, Apr. 2017.
- [2] N. Arefyev, B. Sheludko, A. Podolskiy, and A. Panchenko, “**Always Keep your Target in Mind: Studying Semantics and Improving Performance of Neural Lexical Substitution**,” in *Proceedings of the 28th International Conference on Computational Linguistics*, (Barcelona, Spain (Online)), pp. 1242–1255, International Committee on Computational Linguistics, Dec. 2020.
- [3] A. Kutuzov, M. Dorgham, O. Oliynyk, C. Biemann, and A. Panchenko, “**Making Fast Graph-based Algorithms with Graph Metric Embeddings**,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 3349–3355, Association for Computational Linguistics, July 2019.
- [4] D. Ustalov, N. Arefyev, C. Biemann, and A. Panchenko, “**Negative Sampling Improves Hypernymy Extraction Based on Projection Learning**,” in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, (Valencia, Spain), pp. 543–550, Association for Computational Linguistics, Apr. 2017.
- [5] R. Aly, S. Acharya, A. Ossa, A. Köhn, C. Biemann, and A. Panchenko, “**Every Child Should Have Parents: A Taxonomy Refinement Algorithm Based on Hyperbolic Term Embeddings**,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 4811–4817, Association for Computational Linguistics, July 2019.
- [6] D. Ustalov, A. Panchenko, C. Biemann, and S. P. Ponzetto, “**Watset: Local-Global Graph Clustering with Applications in Sense and Frame Induction**,” *Computational Linguistics*, vol. 45, pp. 423–479, Sept. 2019.
- [7] S. Faralli, A. Panchenko, C. Biemann, and S. P. Ponzetto, “**Linked Disambiguated Distributional Semantic Networks**,” in *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part II* (P. Groth, E. Simperl, A. J. G. Gray, M. Sabou, M. Krötzsch,

F. Lécué, F. Flöck, and Y. Gil, eds.), vol. 9982 of *Lecture Notes in Computer Science*, pp. 56–64, 2016.

- [8] C. Biemann, S. Faralli, A. Panchenko, and S. P. Ponzetto, “**A framework for enriching lexical semantic resources with distributional semantics**,” *Nat. Lang. Eng.*, vol. 24, no. 2, pp. 265–312, 2018.
- [9] D. Ustalov, A. Panchenko, and C. Biemann, “**Watset: Automatic Induction of Synsets from a Graph of Synonyms**,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Vancouver, Canada), pp. 1579–1590, Association for Computational Linguistics, July 2017.
- [10] D. Ustalov, A. Panchenko, A. Kutuzov, C. Biemann, and S. P. Ponzetto, “**Unsupervised Semantic Frame Induction using Triclustering**,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, (Melbourne, Australia), pp. 55–62, Association for Computational Linguistics, July 2018.
- [11] Ö. Sevgili, A. Shelmanov, M. Y. Arkhipov, A. Panchenko, and C. Biemann, “**Neural entity linking: A survey of models based on deep learning**,” *Semantic Web*, vol. 13, no. 3, pp. 527–570, 2022.
- [12] S. Anwar, A. Shelmanov, N. Arefyev, A. Panchenko, and C. Biemann, “**Text augmentation for semantic frame induction and parsing**,” *Language Resources and Evaluation*, vol. 23, no. 3, pp. 527–556, 2023.
- [13] A. Jana, D. Puzyrev, A. Panchenko, P. Goyal, C. Biemann, and A. Mukherjee, “**On the Compositionality Prediction of Noun Phrases using Poincaré Embeddings**,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 3263–3274, Association for Computational Linguistics, July 2019.
- [14] I. Nikishina, V. Logacheva, A. Panchenko, and N. Loukachevitch, “**Studying Taxonomy Enrichment on Diachronic WordNet Versions**,” in *Proceedings of the 28th International Conference on Computational Linguistics*, (Barcelona, Spain (Online)), pp. 3095–3106, International Committee on Computational Linguistics, Dec. 2020.
- [15] I. Nikishina, M. Tikhomirov, V. Logacheva, Y. Nazarov, A. Panchenko, and N. V. Loukachevitch, “**Taxonomy enrichment with text and graph vector representations**,” *Semantic Web*, vol. 13, no. 3, pp. 441–475, 2022.
- [16] S. Faralli, A. Panchenko, C. Biemann, and S. P. Ponzetto, “**The ContrastMedium Algorithm: Taxonomy Induction From Noisy Knowledge Graphs With Just A Few Links**,” in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, (Valencia, Spain), pp. 590–600, Association for Computational Linguistics, Apr. 2017.
- [17] M. Pelevina, N. Arefiev, C. Biemann, and A. Panchenko, “**Making Sense of Word Embeddings**,” in *Proceedings of the 1st Workshop on Representation Learning for*

NLP, (Berlin, Germany), pp. 174–183, Association for Computational Linguistics, Aug. 2016.

- [18] A. Panchenko, F. Marten, E. Ruppert, S. Faralli, D. Ustalov, S. P. Ponzetto, and C. Biemann, “**Unsupervised, Knowledge-Free, and Interpretable Word Sense Disambiguation**,” in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, (Copenhagen, Denmark), pp. 91–96, Association for Computational Linguistics, Sept. 2017.
- [19] A. Panchenko, D. Ustalov, S. Faralli, S. P. Ponzetto, and C. Biemann, “**Improving Hypernymy Extraction with Distributional Semantic Classes**,” in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, (Miyazaki, Japan), European Language Resources Association (ELRA), May 2018.
- [20] V. Logacheva, D. Teslenko, A. Shelmanov, S. Remus, D. Ustalov, A. Kutuzov, E. Artemova, C. Biemann, S. P. Ponzetto, and A. Panchenko, “**Word Sense Disambiguation for 158 Languages using Word Embeddings Only**,” in *Proceedings of the Twelfth Language Resources and Evaluation Conference*, (Marseille, France), pp. 5943–5952, European Language Resources Association, May 2020.
- [21] A. Panchenko, S. Faralli, E. Ruppert, S. Remus, H. Naets, C. Fairon, S. P. Ponzetto, and C. Biemann, “**TAXI at SemEval-2016 Task 13: a Taxonomy Induction Method based on Lexico-Syntactic Patterns, Substrings and Focused Crawling**,” in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, (San Diego, California), pp. 1320–1327, Association for Computational Linguistics, June 2016.
- [22] S. Anwar, A. Shelmanov, A. Panchenko, and C. Biemann, “**Generating Lexical Representations of Frames using Lexical Substitution**,” in *Proceedings of the Probability and Meaning Conference (PaM 2020)*, (Gothenburg), pp. 95–103, Association for Computational Linguistics, June 2020.
- [23] D. Ustalov, A. Panchenko, C. Biemann, and S. P. Ponzetto, “**Unsupervised Sense-Aware Hypernymy Extraction**,” in *Proceedings of the 14th Conference on Natural Language Processing, KONVENS 2018, Vienna, Austria, September 19–21, 2018* (A. Barbaresi, H. Biber, F. Neubarth, and R. Osswald, eds.), pp. 192–201, Österreichische Akademie der Wissenschaften, 2018.
- [24] A. Panchenko, A. Lopukhina, D. Ustalov, K. Lopukhin, N. Arefyev, A. Leontyev, and N. V. Loukachevitch, “**RUSSE’2018: A Shared Task on Word Sense Induction for the Russian Language**,” in *Proceedings of the 24th International Conference on Computational Linguistics and Intellectual Technologies (Dialogue’2018). Moscow, Russia.*, pp. 192–201, RGGU, 2018.
- [25] N. Arefyev, P. Ermolaev, and A. Panchenko, “**How much does a word weigh? Weighting word embeddings for word sense induction**,” in *Proceedings of the 24th International Conference on Computational Linguistics and Intellectual Technologies (Dialogue’2018). Moscow, Russia.*, pp. 201–212, RGGU, 2018.

- [26] D. Ustalov, D. Teslenko, A. Panchenko, M. Chernoskutov, C. Biemann, and S. P. Ponzetto, “**An Unsupervised Word Sense Disambiguation System for Under-Resourced Languages**,” in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, (Miyazaki, Japan), European Language Resources Association (ELRA), May 2018.
- [27] S. Faralli, A. Panchenko, C. Biemann, and S. P. Ponzetto, “**Enriching Frame Representations with Distributionally Induced Senses**,” in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, (Miyazaki, Japan), European Language Resources Association (ELRA), May 2018.
- [28] Ö. Sevgili, A. Panchenko, and C. Biemann, “**Improving Neural Entity Disambiguation with Graph Embeddings**,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, (Florence, Italy), pp. 315–322, Association for Computational Linguistics, July 2019.
- [29] A. Panchenko, “**Best of Both Worlds: Making Word Sense Embeddings Interpretable**,” in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16)*, (Portorož, Slovenia), pp. 2649–2655, European Language Resources Association (ELRA), May 2016.
- [30] D. Ustalov, M. Chernoskutov, C. Biemann, and A. Panchenko, “**Fighting with the Sparsity of Synonymy Dictionaries for Automatic Synset Induction**,” in *Analysis of Images, Social Networks and Texts - 6th International Conference, AIST 2017, Moscow, Russia, July 27-29, 2017, Revised Selected Papers* (W. M. P. van der Aalst, D. I. Ignatov, M. Y. Khachay, S. O. Kuznetsov, V. S. Lempitsky, I. A. Lomazova, N. V. Loukachevitch, A. Napoli, A. Panchenko, P. M. Pardalos, A. V. Savchenko, and S. Wasserman, eds.), vol. 10716 of *Lecture Notes in Computer Science*, pp. 94–105, Springer, 2017.
- [31] A. Panchenko, J. Simon, M. Riedl, and C. Biemann, “**Noun Sense Induction and Disambiguation using Graph-Based Distributional Semantics**,” in *Proceedings of the 13th Conference on Natural Language Processing, KONVENS 2016, Bochum, Germany, September 19-21, 2016* (S. Dipper, F. Neubarth, and H. Zinsmeister, eds.), vol. 16 of *Bochumer Linguistische Arbeitsberichte*, 2016.
- [32] D. Puzyrev, A. Shelmanov, A. Panchenko, and E. Artemova, “**Noun Compositionality Detection Using Distributional Semantics for the Russian Language**,” in *Analysis of Images, Social Networks and Texts - 8th International Conference, AIST 2019, Kazan, Russia, July 17-19, 2019, Revised Selected Papers* (W. M. P. van der Aalst, V. Batagelj, D. I. Ignatov, M. Y. Khachay, V. V. Kuskova, A. Kutuzov, S. O. Kuznetsov, I. A. Lomazova, N. V. Loukachevitch, A. Napoli, P. M. Pardalos, M. Pelillo, A. V. Savchenko, and E. Tutubalina, eds.), vol. 11832 of *Lecture Notes in Computer Science*, pp. 218–229, Springer, 2019.
- [33] N. Arefyev, B. Sheludko, and A. Panchenko, “**Combining Lexical Substitutes in Neural Word Sense Induction**,” in *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, (Varna, Bulgaria), pp. 62–70, INCOMA Ltd., Sept. 2019.

- [34] A. Kutuzov, M. Dorgham, O. Oliynyk, C. Biemann, and A. Panchenko, “**Learning Graph Embeddings from WordNet-based Similarity Measures**,” in *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, (Minneapolis, Minnesota), pp. 125–135, Association for Computational Linguistics, June 2019.
- [35] A. Razzhigaev, N. Arefyev, and A. Panchenko, “**SkoltechNLP at SemEval-2021 Task 2: Generating Cross-Lingual Training Data for the Word-in-Context Task**,” in *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, (Online), pp. 157–162, Association for Computational Linguistics, Aug. 2021.
- [36] D. Puzyrev, A. Shelmanov, A. Panchenko, and E. Artemova, “**A Dataset for Noun Compositionality Detection for a Slavic Language**,” in *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, (Florence, Italy), pp. 56–62, Association for Computational Linguistics, Aug. 2019.
- [37] N. Arefyev, B. Sheludko, A. Davletov, D. Kharchev, A. Nevidomsky, and A. Panchenko, “**Neural GRANNy at SemEval-2019 Task 2: A combined approach for better modeling of semantic relationships in semantic frame induction**,” in *Proceedings of the 13th International Workshop on Semantic Evaluation*, (Minneapolis, Minnesota, USA), pp. 31–38, Association for Computational Linguistics, June 2019.
- [38] S. Anwar, D. Ustalov, N. Arefyev, S. P. Ponzetto, C. Biemann, and A. Panchenko, “**HHMM at SemEval-2019 Task 2: Unsupervised Frame Induction using Contextualized Word Embeddings**,” in *Proceedings of the 13th International Workshop on Semantic Evaluation*, (Minneapolis, Minnesota, USA), pp. 125–129, Association for Computational Linguistics, June 2019.
- [39] A. Panchenko, S. Faralli, S. P. Ponzetto, and C. Biemann, “**Using Linked Disambiguated Distributional Networks for Word Sense Disambiguation**,” in *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications*, (Valencia, Spain), pp. 72–78, Association for Computational Linguistics, Apr. 2017.
- [40] I. Nikishina, I. Andrianov, A. Vakhitova, and A. Panchenko, “**TaxFree: a Visualization Tool for Candidate-free Taxonomy Enrichment**,” in *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing: System Demonstrations*, (Taipei, Taiwan), pp. 39–47, Association for Computational Linguistics, Nov. 2022.
- [41] I. Nikishina, N. Loukachevitch, V. Logacheva, and A. Panchenko, “**Evaluation of Taxonomy Enrichment on Diachronic WordNet Versions**,” in *Proceedings of the 11th Global Wordnet Conference*, (University of South Africa (UNISA)), pp. 126–136, Global Wordnet Association, Jan. 2021.
- [42] I. Nikishina, A. Vakhitova, E. Tutubalina, and A. Panchenko, “**Cross-Modal Contextualized Hidden State Projection Method for Expanding of**

Taxonomic Graphs,” in *Proceedings of TextGraphs-16: Graph-based Methods for Natural Language Processing*, (Gyeongju, Republic of Korea), pp. 11–24, Association for Computational Linguistics, Oct. 2022.

- [43] B. Dorow and D. Widdows, “Discovering Corpus-Specific Word Senses,” in *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics - Volume 2*, EACL ’03, (Budapest, Hungary), pp. 79–82, Association for Computational Linguistics, 2003.
- [44] G. Salton, A. Wong, and C. S. Yang, “A Vector Space Model for Automatic Indexing,” *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.
- [45] J. Véronis, “HyperLex: lexical cartography for information retrieval,” *Computer Speech & Language*, vol. 18, no. 3, pp. 223–252, 2004.
- [46] C. Biemann, “Chinese Whispers: An Efficient Graph Clustering Algorithm and Its Application to Natural Language Processing Problems,” in *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*, TextGraphs-1, (New York, NY, USA), pp. 73–80, Association for Computational Linguistics, 2006.
- [47] D. Hope and B. Keller, “MaxMax: A Graph-Based Soft Clustering Algorithm Applied to Word Sense Induction,” in *Computational Linguistics and Intelligent Text Processing: 14th International Conference, CICLing 2013, Samos, Greece, March 24–30, 2013, Proceedings, Part I*, vol. 7816 of *Lecture Notes in Computer Science*, pp. 368–381, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.
- [48] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” in *Workshop at International Conference on Learning Representations (ICLR)*, (Scottsdale, AZ, USA), pp. 1310–1318, 2013.
- [49] A. Joulin, E. Grave, P. Bojanowski, M. Nickel, and T. Mikolov, “Fast linear model for knowledge graph embeddings,” *arXiv preprint arXiv:1710.10881*, 2017.
- [50] C. Biemann and M. Riedl, “Text: Now in 2D! a framework for lexical expansion with contextual similarity,” *Journal of Language Modelling*, vol. 1, no. 1, pp. 55–95, 2013.
- [51] A. Panchenko, *Similarity measures for semantic relation extraction*. PhD thesis, Université catholique de Louvain, Louvain-la-Neuve, Belgium, Louvain-la-Neuve, Belgium, 2013.
- [52] E. Ruppert, J. Klesy, M. Riedl, and C. Biemann, “Rule-based Dependency Parse Collapsing and Propagation for German and English,” in *Proceedings of the GSCL 2015*, (Duisburg, Germany), pp. 58–66, 2015.
- [53] K. W. Church and P. Hanks, “Word association norms, mutual information, and lexicography,” *Computational linguistics*, vol. 16, no. 1, pp. 22–29, 1990.
- [54] C. Biemann, “Chinese Whispers: an efficient graph clustering algorithm and its application to natural language processing problems,” in *Proceedings of the first workshop on graph based methods for natural language processing*, TextGraphs-1, (New York City, NY, USA), pp. 73–80, Association for Computational Linguistics, 2006.

- [55] C. Biemann, “Co-occurrence Cluster Features for Lexical Substitutions in Context,” in *Proceedings of the 5th Workshop on TextGraphs in conjunction with ACL 2010*, (Uppsala, Sweden), 2010.
- [56] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed Representations of Words and Phrases and their Compositionality,” in *Advances in Neural Information Processing Systems 26*, NIPS 2013, pp. 3111–3119, Harrahs and Harveys, NV, USA: Curran Associates, Inc., 2013.
- [57] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov, “Learning Word Vectors for 157 Languages,” in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, (Miyazaki, Japan), pp. 3483–3487, European Language Resources Association (ELRA), 2018.
- [58] N. Ide and J. Véronis, “Introduction to the special issue on word sense disambiguation: the state of the art,” *Computational linguistics*, vol. 24, no. 1, pp. 2–40, 1998.
- [59] E. Agirre and P. G. Edmonds, *Word sense disambiguation: Algorithms and applications*, vol. 33. Springer Science & Business Media, 2007.
- [60] A. Moro and R. Navigli, “SemEval-2015 Task 13: Multilingual All-Words Sense Disambiguation and Entity Linking,” in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, (Denver, CO, USA), pp. 288–297, Association for Computational Linguistics, 2015.
- [61] R. Navigli, “Word sense disambiguation: A survey,” *ACM Computing Surveys (CSUR)*, vol. 41, no. 2, p. 10, 2009.
- [62] G. A. Miller, “WordNet: a lexical database for English,” *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.
- [63] H. Schütze, “Automatic Word Sense Discrimination,” *Computational Linguistics*, vol. 24, no. 1, pp. 97–123, 1998.
- [64] J. Li and D. Jurafsky, “Do Multi-Sense Embeddings Improve Natural Language Understanding?,” in *Conference on Empirical Methods in Natural Language Processing (EMNLP’2015)*, EMNLP’15, (Lisboa, Portugal), pp. 1722–1732, Association for Computational Linguistics, 2015.
- [65] S. Bartunov, D. Kondrashkin, A. Osokin, and D. Vetrov, “Breaking Sticks and Ambiguities with Adaptive Skip-gram,” in *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS’2016)*, AISTATS’16, (Cadiz, Spain), pp. 130–138, 2016.
- [66] A. Vellido, J. D. Martín, F. Rossi, and P. J. Lisboa, “Seeing is believing: The importance of visualization in real-world machine learning applications,” in *Proceedings of the 19th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN’2011)*, (Bruges, Belgium), pp. 219–226, 2011.
- [67] A. A. Freitas, “Comprehensible classification models: a position paper,” *ACM SIGKDD Explorations Newsletter*, vol. 15, no. 1, pp. 1–10, 2014.

- [68] J. Li, X. Chen, E. Hovy, and D. Jurafsky, “Visualizing and Understanding Neural Models in NLP,” in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (San Diego, CA, USA), pp. 681–691, Association for Computational Linguistics, June 2016.
- [69] C. Biemann, “Turk Bootstrap Word Sense Inventory 2.0: A Large-Scale Resource for Lexical Substitution,” in *Proceedings of the 8th International Conference on Language Resources and Evaluation*, (Istanbul, Turkey), pp. 4038–4042, European Language Resources Association, 2012.
- [70] D. Jurgens and I. Klapaftis, “Semeval-2013 Task 13: Word Sense Induction for Graded and Non-graded Senses,” in *Proceedings of the 2nd Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval’2013)*, vol. 2, (Montreal, Canada), pp. 290–299, Association for Computational Linguistics, 2013.
- [71] S. P. Ponzetto and R. Navigli, “Knowledge-rich word sense disambiguation rivaling supervised systems,” in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL’10, (Uppsala, Sweden), pp. 1522–1531, Association for Computational Linguistics, 2010.
- [72] I. Gurevych, J. Eckle-Kohler, S. Hartmann, M. Matuschek, C. M. Meyer, and C. Wirth, “UBY - A Large-Scale Unified Lexical-Semantic Resource Based on LMF,” in *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL’12, (Avignon, France), pp. 580–590, Association for Computational Linguistics, 2012.
- [73] S. Pavel and J. Euzenat, “Ontology Matching: State of the Art and Future Challenges,” *IEEE Transaction on Knowledge and Data Engineering*, vol. 25, no. 1, pp. 158–176, 2013.
- [74] R. Navigli and S. P. Ponzetto, “BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network,” *Artificial Intelligence*, vol. 193, pp. 217–250, 2012.
- [75] R. Fu, J. Guo, B. Qin, W. Che, H. Wang, and T. Liu, “Learning Semantic Hierarchies via Word Embeddings,” in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Baltimore, MD, USA), pp. 1199–1209, Association for Computational Linguistics, 2014.
- [76] J. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, (Berkeley, California, USA), pp. 281–297, University of California Press, 1967.
- [77] T. Wandmacher, “How semantic is Latent Semantic Analysis?,” in *Proceedings of RÉCITAL 2005*, (Dourdan, France), pp. 525–534, 2005.
- [78] K. Heylen, Y. Peirsman, D. Geeraerts, and D. Speelman, “Modelling Word Similarity: an Evaluation of Automatic Synonymy Extraction Algorithms,” in *Proceedings of the*

Sixth International Conference on Language Resources and Evaluation (LREC'08), LREC 2008, (Marrakech, Morocco), pp. 3243–3249, European Language Resources Association (ELRA), 2008.

- [79] A. Panchenko, “Comparison of the Baseline Knowledge-, Corpus-, and Web-based Similarity Measures for Semantic Relations Extraction,” in *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, (Edinburgh, UK), pp. 11–21, Association for Computational Linguistics, 2011.
- [80] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” *CoRR*, vol. abs/1412.6980, 2014.
- [81] M. Abadi et al., “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems,” *CoRR*, vol. abs/1603.04467, 2016.
- [82] S. Faralli, A. Panchenko, C. Biemann, and S. P. Ponzetto, “Linked Disambiguated Distributional Semantic Networks,” in *The Semantic Web – ISWC 2016: 15th International Semantic Web Conference, Proceedings, Part II*, Lecture Notes in Computer Science, pp. 56–64, Kobe, Japan: Springer International Publishing, 2016.
- [83] C. Biemann, S. Faralli, A. Panchenko, and S. P. Ponzetto, “A framework for enriching lexical semantic resources with distributional semantics,” *Natural Language Engineering*, pp. 1–48, 2018.
- [84] C. Biemann and M. Riedl, “Text: now in 2D! A framework for lexical expansion with contextual similarity,” *Journal of Language Modelling*, vol. 1, no. 1, pp. 55–95, 2013.
- [85] D. Widdows and B. Dorow, “A graph model for unsupervised lexical acquisition,” in *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pp. 1–7, Association for Computational Linguistics, 2002.
- [86] M. Everett and S. P. Borgatti, “Ego network betweenness,” *Social networks*, vol. 27, no. 1, pp. 31–38, 2005.
- [87] A. Ferraresi, E. Zanchetta, M. Baroni, and S. Bernardini, “Introducing and evaluating ukWaC, a very large web-derived corpus of English,” in *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google*, (Marrakech, Morocco), pp. 47–54, European Language Resources Association (ELRA), 2008.
- [88] M. Richter, U. Quasthoff, E. Hallsteinsdóttir, and C. Biemann, “Exploiting the Leipzig Corpora Collection,” in *Proceedings of IS-LTC'06*, (Ljubljana, Slovenia), European Language Resources Association (ELRA), 2006.
- [89] D. Graff and C. Cieri, “English Gigaword corpus,” *Linguistic Data Consortium*, 2003.
- [90] A. Neelakantan, J. Shankar, A. Passos, and A. McCallum, “Efficient Non-parametric Estimation of Multiple Embeddings per Word in Vector Space,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1059–1069, Association for Computational Linguistics, 2014.
- [91] M. Steyvers and J. B. Tenenbaum, “The Large-scale structure of semantic networks: Statistical analyses and a model of semantic growth,” *Cognitive science*, vol. 29, no. 1, pp. 41–78, 2005.

- [92] C. Fellbaum, *WordNet: An Electronic Database*. Cambridge, MA: MIT Press, 1998.
- [93] R. Navigli and S. P. Ponzetto, “BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network,” *Artificial Intelligence*, vol. 193, pp. 217–250, 2012.
- [94] M. Nickel and D. Kiela, “Poincaré Embeddings for Learning Hierarchical Representations,” in *Advances in Neural Information Processing Systems 30*, (Long Beach, CA, USA), pp. 6338–6347, 2017.
- [95] R. Parker, D. Graff, J. Kong, K. Chen, and K. Maeda, “English gigaword forth edition,” in *Linguistic Data Consortium*, (Philadelphia, PA, USA), 2009.
- [96] A. Ferraresi, E. Zanchetta, M. Baroni, and S. Bernardini, “Introducing and evaluating ukWaC, a very large web-derived corpus of English,” in *Proceedings of the 4th Web as Corpus Workshop. Can we beat Google?*, (Marrakech, Morocco), pp. 47–54, 2008.
- [97] D. Goldhahn, T. Eckart, and U. Quasthoff, “Building Large Monolingual Dictionaries at the Leipzig Corpora Collection: From 100 to 200 Languages,” in *Proceedings of the Eight International Conference on Language Resources and Evaluation*, (Istanbul, Turkey), pp. 759–765, 2012.
- [98] A. Panchenko, O. Morozova, and H. Naets, “A semantic similarity measure based on lexico-syntactic patterns,” in *Proceedings of KONVENS 2012*, (Vienna, Austria), pp. 174–178, September 2012.
- [99] J. Seitner, C. Bizer, K. Eckert, S. Faralli, R. M. und Heiko Paulheim, and S. P. Ponzetto, “A Large DataBase of Hypernymy Relations Extracted from the Web,” in *Proceedings of the 10th International Conference on Language Resources and Evaluation*, (Portorož, Slovenia), pp. 360–367, 2016.
- [100] A. Panchenko, S. Faralli, E. Ruppert, S. Remus, H. Naets, C. Fairon, S. P. Ponzetto, and C. Biemann, “TAXI at SemEval-2016 Task 13: a taxonomy Induction Method based on Lexico-syntactic Patterns, Substrings and Focused Crawling,” in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, SemEval@NAACL-HLT’16, (San Diego, CA, USA), pp. 1320–1327, Association for Computational Linguistics, 2016.
- [101] J. Stillwell, *Sources of hyperbolic geometry*. No. History of Mathematics, Volume 10 in 1, American Mathematical Society, 1996.
- [102] V. Khrulkov, L. Mirvakhabova, E. Ustinova, I. Oseledets, and V. Lempitsky, “Hyperbolic Image Embeddings,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [103] R. Fu, J. Guo, B. Qin, W. Che, H. Wang, and T. Liu, “Learning semantic hierarchies via word embeddings,” in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, vol. 1, (Baltimore, MD, USA), pp. 1199–1209, Association for Computational Linguistics, 2014.
- [104] K. Heylen, Y. Peirsman, D. Geeraerts, and D. Speelman, “Modelling word similarity: an evaluation of automatic synonymy extraction algorithms,” in *Proceedings of*

the sixth international language resources and evaluation, (Marrakech, Morocco), pp. 3243–3249, 2008.

- [105] J. Weeds, D. Clarke, J. Reffin, D. Weir, and B. Keller, “Learning to distinguish hypernyms and co-hyponyms,” in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics*, (Dublin, Ireland), pp. 2249–2259, Dublin City University and Association for Computational Linguistics, 2014.
- [106] G. Bordea, P. Buitelaar, S. Faralli, and R. Navigli, “Semeval-2015 task 17: Taxonomy Extraction Evaluation (TExEval),” in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, (Denver, CO, USA), pp. 902–910, 2015.
- [107] R. Tarjan, “Depth first search and linear graph algorithms,” *SIAM Journal on Computing*, vol. 1, no. 2, pp. 146 –160, 1972.
- [108] G. A. Miller, “WordNet: A Lexical Database for English,” *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.
- [109] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, “Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation,” *IEEE Transactions on knowledge and data engineering*, vol. 19, no. 3, pp. 355–369, 2007.
- [110] M. T. Pilehvar and R. Navigli, “From senses to texts: An all-in-one graph-based approach for measuring semantic similarity,” *Artificial Intelligence*, vol. 228, pp. 95–128, 2015.
- [111] B. Lebichot, G. Guex, I. Kivimäki, and M. Saerens, “A Constrained Randomized Shortest-Paths Framework for Optimal Exploration,” *arXiv preprint arXiv:1807.04551*, 2018.
- [112] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” in *Proceedings of the International Conference on Learning Representations (ICLR)*, (San Diego, CA, USA), 2015.
- [113] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed Representations of Words and Phrases and their Compositionality,” in *Advances in Neural Information Processing Systems 26*, (Lake Tahoe, NV, USA), pp. 3111–3119, Curran Associates, Inc., 2013.
- [114] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global Vectors for Word Representation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1532–1543, Association for Computational Linguistics, 2014.
- [115] D. B. Johnson, “Efficient algorithms for shortest paths in sparse networks,” *Journal of the ACM (JACM)*, vol. 24, no. 1, pp. 1–13, 1977.
- [116] A. Amrami and Y. Goldberg, “Word Sense Induction with Neural biLM and Symmetric Patterns,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, (Brussels, Belgium), pp. 4860–4867, Association for Computational Linguistics, Oct.-Nov. 2018.

- [117] T. Schick and H. Schütze, “Rare Words: A Major Problem for Contextualized Embeddings and How to Fix it by Attentive Mimicking.,” in *AAAI*, pp. 8766–8774, 2020.
- [118] O. Melamud, J. Goldberger, and I. Dagan, “context2vec: Learning Generic Context Embedding with Bidirectional LSTM,” in *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, (Berlin, Germany), pp. 51–61, Association for Computational Linguistics, Aug. 2016.
- [119] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep Contextualized Word Representations,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, (New Orleans, Louisiana), pp. 2227–2237, Association for Computational Linguistics, June 2018.
- [120] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.
- [121] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “RoBERTa: A Robustly Optimized BERT Pretraining Approach,” *ArXiv*, vol. abs/1907.11692, 2019.
- [122] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, “Xlnet: Generalized autoregressive pretraining for language understanding,” in *Advances in neural information processing systems*, pp. 5753–5763, 2019.
- [123] N. Arefyev, B. Sheludko, and A. Panchenko, “Combining Lexical Substitutes in Neural Word Sense Induction,” in *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP’19)*, RANLP ’19, (Varna, Bulgaria), pp. 62–70, 2019.

Appendix A

Copies of published research papers and journal articles

A.1 Paper “Unsupervised Does Not Mean Uninterpretable: The Case for Word Sense Induction and Disambiguation”

A. Panchenko, E. Ruppert, S. Faralli, S. P. Ponzetto, and C. Biemann, “**Unsupervised does not mean uninterpretable: The case for word sense induction and disambiguation**,” in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, (Valencia, Spain), pp. 86–98, Association for Computational Linguistics, Apr. 2017

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/E17-1009>

Unsupervised Does Not Mean Uninterpretable: The Case for Word Sense Induction and Disambiguation

Alexander Panchenko[‡], Eugen Ruppert[‡], Stefano Faralli[†],
Simone Paolo Ponzetto[†] and Chris Biemann[‡]

[‡]Language Technology Group, Computer Science Dept., University of Hamburg, Germany

[†]Web and Data Science Group, Computer Science Dept., University of Mannheim, Germany

{panchenko, ruppert, biemann}@informatik.uni-hamburg.de

{faralli, simone}@informatik.uni-mannheim.de

Abstract

The current trend in NLP is the use of highly opaque models, e.g. neural networks and word embeddings. While these models yield state-of-the-art results on a range of tasks, their drawback is poor interpretability. On the example of word sense induction and disambiguation (WSID), we show that it is possible to develop an interpretable model that matches the state-of-the-art models in accuracy. Namely, we present an unsupervised, knowledge-free WSID approach, which is interpretable at three levels: word sense inventory, sense feature representations, and disambiguation procedure. Experiments show that our model performs on par with state-of-the-art word sense embeddings and other unsupervised systems while offering the possibility to justify its decisions in human-readable form.

1 Introduction

A word sense disambiguation (WSD) system takes as input a target word t and its context C . The system returns an identifier of a word sense s_i from the word sense inventory $\{s_1, \dots, s_n\}$ of t , where the senses are typically defined manually in advance. Despite significant progress in methodology during the two last decades (Ide and Véronis, 1998; Agirre and Edmonds, 2007; Moro andNavigli, 2015), WSD is still not widespread in applications (Navigli, 2009), which indicates the need for further progress. The difficulty of the problem largely stems from the lack of domain-specific training data. A fixed sense inventory, such as the one of WordNet (Miller, 1995), may contain irrelevant senses for the given application and at the same time lack relevant domain-specific senses.

Word sense induction from domain-specific corpora is a supposed to solve this problem. However, most approaches to word sense induction and disambiguation, e.g. (Schütze, 1998; Li and Jurafsky, 2015; Bartunov et al., 2016), rely on clustering methods and dense vector representations that make a WSD model uninterpretable as compared to knowledge-based WSD methods.

Interpretability of a statistical model is important as it lets us understand the reasons behind its predictions (Vellido et al., 2011; Freitas, 2014; Li et al., 2016). Interpretability of WSD models (1) lets a user understand why in the given context one observed a given sense (e.g., for educational applications); (2) performs a comprehensive analysis of correct and erroneous predictions, giving rise to improved disambiguation models.

The contribution of this paper is an interpretable unsupervised knowledge-free WSD method. The novelty of our method is in (1) a technique to disambiguation that relies on induced inventories as a pivot for learning sense feature representations, (2) a technique for making induced sense representations interpretable by labeling them with hypernyms and images.

Our method tackles the interpretability issue of the prior methods; it is interpretable at the levels of (1) sense inventory, (2) sense feature representation, and (3) disambiguation procedure. In contrast to word sense induction by context clustering (Schütze (1998), *inter alia*), our method constructs an explicit word sense inventory. The method yields performance comparable to the state-of-the-art unsupervised systems, including two methods based on word sense embeddings. An open source implementation of the method featuring a live demo of several pre-trained models is available online.¹

¹<http://www.jobimtext.org/wsd>

2 Related Work

Multiple designs of WSD systems were proposed (Agirre and Edmonds, 2007; Navigli, 2009). They vary according to the level of supervision and the amount of external knowledge used. Most current systems either make use of lexical resources and/or rely on an explicitly annotated sense corpus.

Supervised approaches use a sense-labeled corpus to train a model, usually building one submodel per target word (Ng, 1997; Lee and Ng, 2002; Klein et al., 2002; Wee, 2010). The IMS system by Zhong and Ng (2010) provides an implementation of the supervised approach to WSD that yields state-of-the-art results. While supervised approaches demonstrate top performance in competitions, they require large amounts of sense-labeled examples per target word.

Knowledge-based approaches rely on a lexical resource that provides a sense inventory and features for disambiguation and vary from the classical Lesk (1986) algorithm that uses word definitions to the Babelfy (Moro et al., 2014) system that uses harnesses a multilingual lexical-semantic network. Classical examples of such approaches include (Banerjee and Pedersen, 2002; Pedersen et al., 2005; Miller et al., 2012). More recently, several methods were proposed to learn sense embeddings on the basis of the sense inventory of a lexical resource (Chen et al., 2014; Rothe and Schütze, 2015; Camacho-Collados et al., 2015; Iacobacci et al., 2015; Nieto Piña and Johansson, 2016).

Unsupervised knowledge-free approaches use neither handcrafted lexical resources nor hand-annotated sense-labeled corpora. Instead, they induce word sense inventories automatically from corpora. Unsupervised WSD methods fall into two main categories: context clustering and word ego-network clustering.

Context clustering approaches, e.g. (Pedersen and Bruce, 1997; Schütze, 1998), represent an instance usually by a vector that characterizes its context, where the definition of context can vary greatly. These vectors of each instance are then clustered. Multi-prototype extensions of the skip-gram model (Mikolov et al., 2013) that use no pre-defined sense inventory learn one embedding word vector per one word sense and are commonly fitted with a disambiguation mechanism (Huang et al., 2012; Tian et al., 2014; Neelakantan et al.,

2014; Bartunov et al., 2016; Li and Jurafsky, 2015; Pelevina et al., 2016). Comparisons of the *AdaGram* (Bartunov et al., 2016) to (Neelakantan et al., 2014) on three SemEval word sense induction and disambiguation datasets show the advantage of the former. For this reason, we use *AdaGram* as a representative of the state-of-the-art word sense embeddings in our experiments. In addition, we compare to SenseGram, an alternative sense embedding based approach by Pelevina et al. (2016). What makes the comparison to the later method interesting is that this approach is similar to ours, but instead of sparse representations the authors rely on word embeddings, making their approach less interpretable.

Word ego-network clustering methods (Lin, 1998; Pantel and Lin, 2002; Widdows and Dorow, 2002; Biemann, 2006; Hope and Keller, 2013) cluster graphs of words semantically related to the ambiguous word. An ego network consists of a single node (ego) together with the nodes they are connected to (alters) and all the edges among those alters (Everett and Borgatti, 2005). In our case, such a network is a local neighborhood of one word. Nodes of the ego-network can be (1) words semantically similar to the target word, as in our approach, or (2) context words relevant to the target, as in the *UoS* system (Hope and Keller, 2013). Graph edges represent semantic relations between words derived using corpus-based methods (e.g. distributional semantics) or gathered from dictionaries. The sense induction process using word graphs is explored by (Widdows and Dorow, 2002; Biemann, 2006; Hope and Keller, 2013). Disambiguation of instances is performed by assigning the sense with the highest overlap between the instance's context words and the words of the sense cluster. Véronis (2004) compiles a corpus with contexts of polysemous nouns using a search engine. A word graph is built by drawing edges between co-occurring words in the gathered corpus, where edges below a certain similarity threshold were discarded. His HyperLex algorithm detects hubs of this graph, which are interpreted as word senses. Disambiguation in this experiment is performed by computing the distance between context words and hubs in this graph.

Di Marco and Navigli (2013) presents a comprehensive study of several graph-based WSI methods including Chinese Whispers, HyperLex, curvature clustering (Dorow et al., 2005). Besides,

authors propose two novel algorithms: Balanced Maximum Spanning Tree Clustering and Squares (B-MST), Triangles and Diamonds (SquaT++). To construct graphs, authors use first-order and second-order relations extracted from a background corpus as well as keywords from snippets. This research goes beyond intrinsic evaluations of induced senses and measures impact of the WSI in the context of an information retrieval via clustering and diversifying Web search results. Depending on the dataset, HyperLex, B-MST or Chinese-Whispers provided the best results.

Our system combines several of above ideas and adds features ensuring interpretability. Most notably, we use a word sense inventory based on clustering word similarities (Pantel and Lin, 2002); for disambiguation we rely on syntactic context features, co-occurrences (Hope and Keller, 2013) and language models (Yuret, 2012).

Interpretable approaches. The need in methods that interpret results of opaque statistical models is widely recognised (Vellido et al., 2011; Vellido et al., 2012; Freitas, 2014; Li et al., 2016; Park et al., 2016). An interpretable WSD system is expected to provide (1) a human-readable sense inventory, (2) human-readable reasons why in a given context c a given sense s_i was detected. Lexical resources, such as WordNet, solve the first problem by providing manually-crafted definitions of senses, examples of usage, hypernyms, and synonyms. The BabelNet (Navigli and Ponzetto, 2010) integrates all these sense representations, adding to them links to external resources, such as Wikipedia, topical category labels, and images representing the sense. The unsupervised models listed above do not feature any of these representations making them much less interpretable as compared to the knowledge-based models. Ruppert et al. (2015) proposed a system for visualising sense inventories derived in an unsupervised way using graph-based distributional semantics. Panchenko (2016) proposed a method for making sense inventory of word sense embeddings interpretable by mapping it to BabelNet.

Our approach was inspired by the knowledge-based system Babelfy (Moro et al., 2014). While the inventory of Babelfy is interpretable as it relies on BabelNet, the system provides no underlying reasons behind sense predictions. Our objective was to reach interpretability level of knowledge-based models within an unsupervised framework.

3 Method: Unsupervised Interpretable Word Sense Disambiguation

Our unsupervised word sense disambiguation method consist of the five steps illustrated in Figure 1: extraction of context features (Section 3.1); computing word and feature similarities (Section 3.2); word sense induction (Section 3.3); labeling of clusters with hypernyms and images (Section 3.4), disambiguation of words in context based on the induced inventory (Section 3.5), and finally interpretation of the model (Section 3.6). Feature similarity and co-occurrence computation steps (drawn with a dashed lines) are optional, since they did not consistently improve performance.

3.1 Extraction of Context Features

The goal of this step is to extract word-feature counts from the input corpus. In particular, we extract three types of features:

Dependency Features. These feature represents a word by a syntactic dependency such as “nn(•,writing)” or “prep.at(sit,•)”, extracted from the Stanford Dependencies (De Marneffe et al., 2006) obtained with the the PCFG model of the Stanford parser (Klein and Manning, 2003). Weights are computed using the Local Mutual Information (LMI) (Evert, 2005). One word is represented with 1000 most significant features.

Co-occurrence Features. This type of features represents a word by another word. We extract the list of words that significantly co-occur in a sentence with the target word in the input corpus based on the log-likelihood as word-feature weight (Dunning, 1993).

Language Model Feature. This type of features are based on a trigram model with Kneser-Ney smoothing (Kneser and Ney, 1995). In particular, a word is represented by (1) right and left context words, e.g. “office_•_and”, (2) two preceding words “new_office_•”, and (3) two succeeding words, e.g. “•_and_chairs”. We use the conditional probabilities of the resulting trigrams as word-feature weights.

3.2 Computing Word and Feature Similarities

The goal of this step is to build a graph of word similarities, such as (table, chair, 0.78). We used the *JoBimText* framework (Biemann and Riedl,

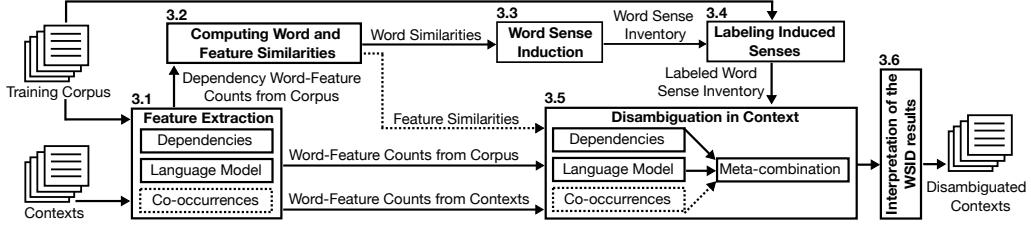


Figure 1: Outline of our unsupervised interpretable method for word sense induction and disambiguation.

2013) as it yields comparable performance on semantic similarity to state-of-the-art dense representations (Mikolov et al., 2013) compared on the WordNet as gold standard (Riedl, 2016), but is interpretable as words are represented by sparse interpretable features. Namely we use dependency-based features as, according to prior evaluations, this kind of features provides state-of-the-art semantic relatedness scores (Padó and Lapata, 2007; Van de Cruys, 2010; Panchenko and Morozova, 2012; Levy and Goldberg, 2014).

First, features of each word are ranked using the LMI metric (Evert, 2005). Second, the word representations are pruned keeping 1000 most salient features per word and 1000 most salient words per feature. The pruning reduces computational complexity and noise. Finally, word similarities are computed as a number of common features for two words. This is again followed by a pruning step in which only the 200 most similar terms are kept to every word. The resulting word similarities are browsable online.²

Note that while words can be characterized with distributions over features, features can vice versa be characterized by a distribution over words. We use this duality to compute feature similarities using the same mechanism and explore their use in disambiguation below.

3.3 Word Sense Induction

We induce a sense inventory by clustering of ego-network of similar words. In our case, an inventory represents senses by a word cluster, such as “chair, bed, bench, stool, sofa, desk, cabinet” for the “furniture” sense of the word “table”.

The sense induction processes one word t of the distributional thesaurus T per iteration. First, we retrieve nodes of the ego-network G of t being the N most similar words of t according to T (see

²Select the “JoBimViz” demo and then the “Stanford (English)” model: <http://www.jobimtext.org>.

Figure 2 (1)). Note that the target word t itself is not part of the ego-network. Second, we connect each node in G to its n most similar words according to T . Finally, the ego-network is clustered with Chinese Whispers (Biemann, 2006), a non-parametric algorithm that discovers the number of senses automatically. The n parameter regulates the granularity of the inventory: we experiment with $n \in \{200, 100, 50\}$ and $N = 200$.

The choice of Chinese Whispers among other algorithms, such as HyperLex (Véronis, 2004) or MCL (Van Dongen, 2008), was motivated by the absence of meta-parameters and its comparable performance on the WSI task to the state-of-the-art (Di Marco and Navigli, 2013).

3.4 Labeling Induced Senses with Hyponyms and Images

Each sense cluster is automatically labeled to improve its interpretability. First, we extract hyponyms from the input corpus using Hearst (1992) patterns. Second, we rank hyponyms relevant to the cluster by a product of two scores: the *hyponym relevance* score, calculated as $\sum_{w \in \text{cluster}} \text{sim}(t, w) \text{freq}(w, h)$, and the *hyponym coverage* score, calculated as $\sum_{w \in \text{cluster}} \min(\text{freq}(w, h), 1)$. Here the $\text{sim}(t, w)$ is the relatedness of the cluster word w to the target word t , and the $\text{freq}(w, h)$ is the frequency of the hyponymy relation (w, h) as extracted via patterns. Thus, a high-ranked hyponym h has high relevance, but also is confirmed by several cluster words. This stage results in a ranked list of labels that specify the word sense, for which we here show the first one, e.g. “table (furniture)” or “table (data)”.

Faralli and Navigli (2012) showed that web search engines can be used to bootstrap sense-related information. To further improve interpretability of induced senses, we assign an image to each word in the cluster (see Figure 2) by query-

ing the Bing image search API³ using the query composed of the target word and its hypernym, e.g. “jaguar car”. The first hit of this query is selected to represent the induced word sense.

Algorithm 1: Unsupervised WSD of the word t based on the induced word sense inventory I .

```

input : Word  $t$ , context features  $C$ , sense inventory  $I$ ,
        word-feature table  $F$ , use largest cluster
        back-off  $LCB$ , use feature expansion  $FE$ .
output: Sense of the target word  $t$  in inventory  $I$  and
        confidence score.

1  $S \leftarrow \text{getSenses}(I, t)$ 
2 if  $FE$  then
3   |  $C \leftarrow \text{featureExpansion}(C)$ 
4 end
5 foreach  $(sense, cluster) \in S$  do
6   |  $\alpha[sense] \leftarrow \{\}$ 
7   | foreach  $w \in cluster$  do
8     |   | foreach  $c \in C$  do
9       |     |   |  $\alpha[sense] \leftarrow \alpha[sense] \cup F(w, c)$ 
10      |   | end
11    | end
12 end
13 if  $\max_{sense \in S} \text{mean}(\alpha[sense]) = 0$  then
14   |   | if  $LCB$  then
15     |     |   | return  $\arg \max_{(., cluster) \in S} |cluster|$ 
16   |   | else
17     |     |   | return  $-1 // \text{reject to classify}$ 
18   | end
19 else
20   |   | return  $\arg \max_{(sense, .) \in S} \text{mean}(\alpha[sense])$ 
21 end

```

3.5 Word Sense Disambiguation with Induced Word Sense Inventory

To disambiguate a target word t in context, we extract context features C and pass them to Algorithm 1. We use the induced sense inventory I and select the sense that has the largest weighted feature overlap with context features or fall back to the largest cluster back-off when context features C do not match the learned sense representations.

The algorithm starts by retrieving induced sense clusters of the target word (line 1). Next, the method starts to accumulate context feature weights of each *sense* in $\alpha[sense]$. Each word w in a sense *cluster* brings all its word-feature counts $F(w, c)$: see lines 5-12. Finally, a *sense* that maximizes mean weight across all context features is chosen (lines 13-21). Optionally, we can resort to the largest cluster back-off (LCB) strategy in case if no context features match sense representations.

³<https://azure.microsoft.com/en-us/services/cognitive-services/search>

Note that the induced inventory I is used as a pivot to aggregate word-feature counts $F(w, c)$ of the words in the *cluster* in order to build feature representations of each induced *sense*. We assume that the sets of similar words per sense are compatible with each other’s context. Thus, we can aggregate ambiguous feature representations of words in a sense cluster. In a way, occurrences of cluster members form the training set for the sense, i.e. contexts of {chair, bed, bench, stool, sofa, desk, cabinet}, add to the representation of “table (furniture)” in the model. Here, ambiguous cluster members like “chair” (which could also mean “chairman”) add some noise, but its influence is dwarfed by the aggregation over all cluster members. Besides, it is unlikely that the target (“table”) and the cluster member (“chair”) share the same homonymy, thus noisy context features hardly play a role when disambiguating the target in context. For instance, for scoring using language model features, we retrieve the context of the target word and substitute the target word one by one of the cluster words. To close the gap between the aggregated dependency per sense $\alpha[sense]$ and dependencies observed in the target’s context C , we use the similarity of features: we expand every feature $c \in C$ with 200 of most similar features and use them as additional features (lines 2-4).

We run disambiguation independently for each of the feature types listed above, e.g. dependencies or co-occurrences. Next, independent predictions are combined using the majority-voting rule.

3.6 Interpretability of the Method

Results of disambiguation can be interpreted by humans as illustrated by Figure 2. In particular, our approach is interpretable at three levels:

1. Word sense inventory. To make induced word sense inventories interpretable we display senses of each word as an ego-network of its semantically related words. For instance, the network of the word “table” in our example is constructed from two tightly related groups of words that correspond to “furniture” and “data” senses. These labels of the clusters are obtained automatically (see Section 3.4).

While alternative methods, such as *AdaGram*, can generate sense clusters, our approach makes the senses better interpretable due to hypernyms and image labels that summarize senses.

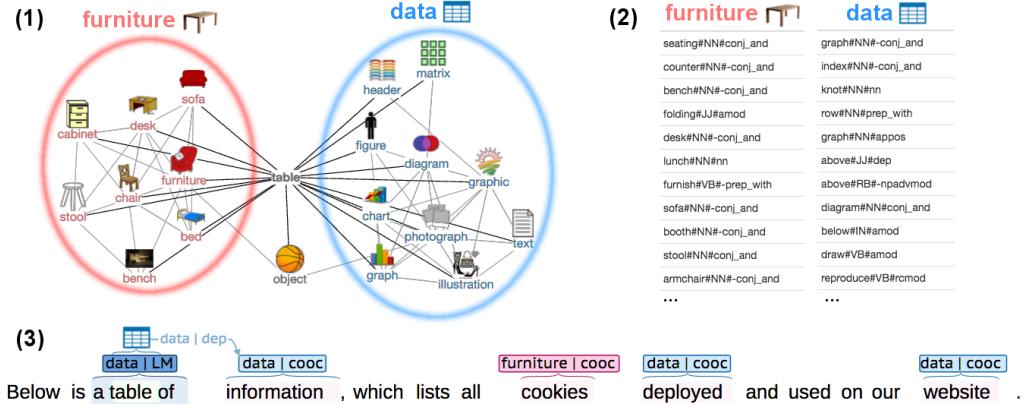


Figure 2: Interpretation of the senses of the word “table” at three levels by our method: (1) word sense inventory; (2) sense feature representation; (3) results of disambiguation in context. The sense labels (“furniture” and “data”) are obtained automatically based on cluster labeling with hypernyms. The images associated with the senses are retrieved using a search engine: “table data” and “table furniture”.

2. Sense feature representation. Each sense in our model is characterized by a list of sparse features ordered by relevance to the sense. Figure 2 (2) shows most salient dependency features to senses of the word “table”. These feature representations are obtained by aggregating features of sense cluster words.

In systems based on dense vector representations, there is no straightforward way to get the most salient features of a sense, which makes the analysis of learned representations problematic.

3. Disambiguation method. To provide the reasons for sense assignment in context, our method highlights the most discriminative context features that caused the prediction. The discriminative power of a feature is defined as the ratio between its weights for different senses.

In Figure 2 (3) words “information”, “cookies”, “deployed” and “website” are highlighted as they are most discriminative and intuitively indicate on the “data” sense of the word “table” as opposed to the “furniture” sense. The same is observed for other types of features. For instance, the syntactic dependency to the word “information” is specific to the “data” sense.

Alternative unsupervised WSD methods that rely on word sense embeddings make it difficult to explain sense assignment in context due to the use of dense features whose dimensions are not interpretable.

4 Experiments

We use two lexical sample collections suitable for evaluation of unsupervised WSD systems. The first one is the Turk Bootstrap Word Sense Inventory (TWSI) dataset introduced by Biemann (2012). It is used for testing different configurations of our approach. The second collection, the SemEval 2013 word sense induction dataset by Jurgens and Klapaftis (2013), is used to compare our approach to existing systems. In both datasets, to measure WSD performance, induced senses are mapped to gold standard senses. In experiments with the TWSI dataset, the models were trained on the Wikipedia corpus⁴ while in experiments with the SemEval datasets models are trained on the ukWaC corpus (Baroni et al., 2009) for a fair comparison with other participants.

4.1 TWSI Dataset

4.1.1 Dataset and Evaluation Metrics

This test collection is based on a crowdsourced resource that comprises 1,012 frequent nouns with 2,333 senses and average polysemy of 2.31 senses per word. For these nouns, 145,140 annotated sentences are provided. Besides, a sense inventory is explicitly provided, where each sense is represented with a list of words that can substitute target noun in a given sentence. The sense distribution across sentences in the dataset is highly

⁴We use a Wikipedia dump from September 2015: <http://doi.org/10.5281/zenodo.229904>

skewed as 79% of contexts are assigned to the most frequent senses. Thus, in addition to the full TWSI dataset, we also use a balanced subset featuring five contexts per sense and 6,166 sentences to assess the quality of the disambiguation mechanism for smaller senses. This dataset contains no monosemous words to completely remove the bias of the most frequent sense. Note that de-biasing the evaluation set does not de-bias the word sense inventory, thus the task becomes harder for the balanced subset.

For the TWSI evaluation, we create an explicit mapping between the system-provided sense inventory and the TWSI word senses: senses are represented as the bag of words, which are compared using cosine similarity. Every induced sense gets assigned at most one TWSI sense. Once the mapping is completed, we calculate Precision, Recall, and F-measure. We use the following baselines to facilitate interpretation of the results: (1) MFS of the TWSI inventory always assigns the most frequent sense in the TWSI dataset; (2) LCB of the induced inventory always assigns the largest sense cluster; (3) Upper bound of the induced vocabulary always selects the correct sense for the context, but only if the mapping exists for this sense; (4) Random sense of the TWSI and the induced inventories.

4.1.2 Discussion of Results

The results of the TWSI evaluation are presented in Table 1. In accordance with prior art in word sense disambiguation, the most frequent sense (MFS) proved to be a strong baseline, reaching an F-score of 0.787, while the random sense over the TWSI inventory drops to 0.536. The upper bound on our induced inventory (F-score of 0.900) shows that the sense mapping technique used prior to evaluation does not drastically distort the evaluation scores. The LCB baseline of the induced inventory achieves an F-score of 0.691, demonstrating the efficiency of the LCB technique.

Let us first consider models based on single features. Dependency features yield the highest precision of 0.728, but have a moderate recall of 0.343 since they rarely match due to their sparsity. The LCB strategy for these rejected contexts helps to improve recall at cost of precision. Co-occurrence features yield significantly lower precision than the dependency-based features, but their recall is higher. Finally, the language model features yield very balanced results in terms of

both precision and recall. Yet, the precision of the model based on this feature type is significantly lower than that of dependencies.

Not all combinations improve results, e.g. combination of three types of features yields inferior results as compared to the language model alone. However, a combination of the language model with dependency features does provide an improvement over the single models as both these models bring strong signal of complementary nature about the semantics of the context. The dependency features represent syntactic information, while the LM features represent lexical information. This improvement is even more pronounced in the case of the balanced TWSI dataset. This combined model yields the best F-scores overall.

Table 2 presents the effect of the feature expansion based on the graph of similar features. For a low-recall model such the one based on syntactic dependencies, feature expansion makes a lot of sense: it almost doubles recall, while losing some precision. The gain in F-score using this technique is almost 20 points on the full TWSI dataset. However, the need for such expansion vanishes when two principally different types of features (precise syntactic dependencies and high-coverage trigram language model) are combined. Both precision and F-score of this combined model outperforms that of the dependency-based model with feature expansion by a large margin.

Figure 3 illustrates how granularity of the induced sense inventory influences WSD performance. For this experiment, we constructed three inventories, setting the number of most similar words in the ego-network n to 200, 100 and 50. These settings produced inventories with respectively 1.96, 2.98 and 5.21 average senses per target word. We observe that a higher sense granularity leads to lower F-scores. This can be explained because of (1) the fact that granularity of the TWSI is similar to granularity of the most coarse-grained inventory; (2) the higher the number of senses, the higher the chance to make a wrong sense assignment; (3) due to the reduced size of individual clusters, we get less signal per sense cluster and noise becomes more pronounced.

To summarize, the best precision is reached by a model based on un-expanded dependencies and the best F-score can be obtained by a combination of models based on un-expanded dependency features and language model features.

Model	#Senses	Full TWSI			Sense-Balanced TWSI		
		Prec.	Recall	F-score	Prec.	Recall	F-score
MFS of the TWSI inventory	2.31	0.787	0.787	0.787	0.373	0.373	0.373
Random Sense of the TWSI inventory	2.31	0.536	0.534	0.535	0.160	0.160	0.160
Upper bound of the induced inventory	1.96	1.000	0.819	0.900	1.000	0.598	0.748
Largest Cluster Back-Off (LCB) of the induced inventory	1.96	0.691	0.690	0.691	0.371	0.371	0.371
Random sense of the induced inventory	1.96	0.559	0.558	0.558	0.325	0.324	0.324
Dependencies	1.96	0.728	0.343	0.466	0.432	0.190	0.263
Dependencies + LCB	1.96	0.689	0.680	0.684	0.388	0.385	0.387
Co-occurrences (Cooc)	1.96	0.570	0.563	0.566	0.336	0.333	0.335
Language Model (LM)	1.96	0.685	0.677	0.681	0.416	0.412	0.414
Dependencies + LM + Cooc	1.96	0.644	0.636	0.640	0.388	0.386	0.387
Dependencies + LM	1.96	0.689	0.681	0.685	0.426	0.422	0.424

Table 1: WSD performance of different configurations of our method on the full and the sense-balanced TWSI datasets based on the coarse inventory with 1.96 senses/word ($N = 200, n = 200$).

Model	Precision	Recall	F-score	Precision	Recall	F-score
Dependencies	0.728	0.343	0.466	0.432	0.190	0.263
Dependencies Exp.	0.687	0.633	0.659	0.414	0.379	0.396
Dependencies + LM	0.689	0.681	0.685	0.426	0.422	0.424
Dependencies Exp. + LM	0.684	0.676	0.680	0.412	0.408	0.410

Table 2: Effect of the feature expansion: performance on the full (on the left) and the sense-balanced (on the right) TWSI datasets. The models were trained on the Wikipedia corpus using the coarse inventory (1.96 senses per word). The best results overall are underlined.

4.2 SemEval 2013 Task 13 Dataset

4.2.1 Dataset and Evaluation Metrics

The task of word sense induction for graded and non-graded senses provides 20 nouns, 20 verbs and 10 adjectives in WordNet-sense-tagged contexts. It contains 20-100 contexts per word, and 4,664 contexts in total with 6,73 sense per word in average. Participants were asked to cluster instances into groups corresponding to distinct word senses. Instances with multiple senses were labeled with a score between 0 and 1.

Performance is measured with three measures that require a mapping of inventories (Jaccard Index, Tau, WNDCG) and two cluster comparison measures (Fuzzy NMI, Fuzzy B-Cubed).

4.2.2 Discussion of Results

Table 3 presents results of evaluation of the best configuration of our approach trained on the ukWaC corpus. We compare our approach to four SemEval participants and two state-of-the-art systems based on word sense embeddings: *AdaGram* (Bartunov et al., 2016) based on Bayesian stick-breaking process⁵ and *SenseGram* (Pelevina et al., 2016) based on clustering of ego-network

generated using word embeddings⁶. The *AI-KU* system (Baskaya et al., 2013) directly clusters test contexts using the k -means algorithm based on lexical substitution features. The *Unimelb* system (Lau et al., 2013) uses one hierarchical topic model to induce and disambiguate senses of one word. The *UoS* system (Hope and Keller, 2013) induces senses by building an ego-network of a word using dependency relations, which is subsequently clustered using the MaxMax clustering algorithm. The *La Sapienza* system (Jurgens and Klapaftis, 2013), relies on WordNet for the sense inventory and disambiguation.

In contrast to the TWSI evaluation, the most fine-grained model yields the best scores, yet the inventory of the task is also more fine-grained than the one of the TWSI (7.08 vs. 2.31 avg. senses per word). Our method outperforms the knowledge-based system of *La Sapienza* according to two of three metrics metrics and the *SenseGram* system based on sense embeddings according to four of five metrics. Note that *SenseGram* outperforms all other systems according to the Fuzzy B-Cubed metric, which is maximized in the “All instances, One sense” settings. Thus this result may be due to

⁵<https://github.com/sbos/AdaGram.jl>

⁶<https://github.com/tudarmstadt-lt/sensemgram>

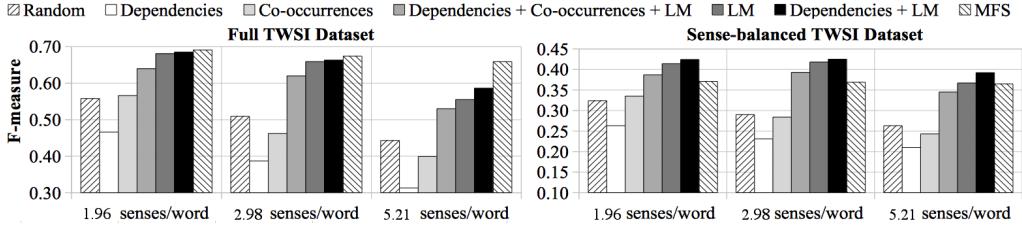


Figure 3: Impact of word sense inventory granularity on WSD performance: the TWSI dataset.

Model	Jacc. Ind.	Tau	WNDCG	Fuzzy NMI	Fuzzy B-Cubed
All Instances, One sense	0.192	0.609	0.288	0.000	0.623
1 sense per instance	0.000	0.953	0.000	0.072	0.000
Most Frequent Sense	0.552	0.560	0.412	–	–
AI-KU	0.197	0.620	0.387	0.065	0.390
AI-KU (remove5-add1000)	0.245	0.642	0.332	0.039	0.451
Unimelb (50k)	0.213	0.620	0.371	0.060	0.483
UoS (top-3)	0.232	0.625	0.374	0.045	0.448
La Sapienza (2)	0.149	0.510	0.383	–	–
AdaGram, $\alpha = 0.05$, 100 dim. vectors	0.274	0.644	0.318	0.058	0.470
SenseGram, 100 dim., CBOW, weight, sim., $p = 2$	0.197	0.615	0.291	0.011	0.615
Dependencies + LM (1.96 senses/word)	0.239	0.634	0.300	0.041	0.513
Dependencies + LM (2.98 senses/word)	0.242	0.634	0.300	0.041	0.504
Dependencies + LM (5.21 senses/word)	0.253	0.638	0.300	0.041	0.479

Table 3: WSD performance of the best configuration of our method identified on the TWSI dataset as compared to participants of the SemEval 2013 Task 13 and two systems based on word sense embeddings (AdaGram and SenseGram). All models were trained on the ukWaC corpus.

difference in granularities: the average polysemy of the *SenseGram* model is 1.56, while the polysemy of our models range from 1.96 to 5.21.

Besides, our system performs comparably to the top unsupervised systems participated in the competition: It is on par with the top SemEval submissions (*AI-KU* and *UoS*) and the another system based on embeddings (*AdaGram*), in terms of four out of five metrics (Jaccard Index, Tau, Fuzzy B-Cubed, Fuzzy NMI).

Therefore, we conclude that our system yields comparable results to the state-of-the-art unsupervised systems. Note, however, that none of the rivaling systems has a comparable level of interpretability to our approach. This is where our method is unique in the class of unsupervised methods: feature representations and disambiguation procedure of the neural-based *AdaGram* and *SenseGram* systems cannot be straightforwardly interpreted. Besides, inventories of the existing systems are represented as ranked lists of words lacking features that improve readability, such as hypernyms and images.

5 Conclusion

In this paper, we have presented a novel method for word sense induction and disambiguation that relies on a meta-combination of dependency features with a language model. The majority of existing unsupervised approaches focus on optimizing the accuracy of the method, sacrificing its interpretability due to the use of opaque models, such as neural networks. In contrast, our approach places a focus on interpretability with the help of sparse readable features. While being interpretable at three levels (sense inventory, sense representations and disambiguation), our method is competitive to the state-of-the-art, including two recent approaches based on sense embeddings, in a word sense induction task. Therefore, it is possible to match the performance of accurate, but opaque methods when interpretability matters.

Acknowledgments

We acknowledge the support of the Deutsche Forschungsgemeinschaft (DFG) foundation under the JOIN-T project.

References

- Eneko Agirre and Philip G. Edmonds. 2007. *Word sense disambiguation: Algorithms and applications*, volume 33. Springer Science & Business Media.
- Satanjeev Banerjee and Ted Pedersen. 2002. An adapted Lesk algorithm for word sense disambiguation using WordNet. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics*, pages 136–145, Mexico City, Mexico. Springer.
- Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3):209–226.
- Sergey Bartunov, Dmitry Kondrashkin, Anton Osokin, and Dmitry Vetrov. 2016. Breaking sticks and ambiguities with adaptive skip-gram. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS'2016)*, Cadiz, Spain.
- Osman Baskaya, Enis Sert, Volkan Cirik, and Deniz Yuret. 2013. AI-KU: Using Substitute Vectors and Co-Occurrence Modeling for Word Sense Induction and Disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 300–306, Atlanta, GA, USA. Association for Computational Linguistics.
- Chris Biemann and Martin Riedl. 2013. Text: Now in 2D! a framework for lexical expansion with contextual similarity. *Journal of Language Modelling*, 1(1):55–95.
- Chris Biemann. 2006. Chinese Whispers: an efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of the first workshop on graph based methods for natural language processing*, pages 73–80, New York City, NY, USA. Association for Computational Linguistics.
- Chris Biemann. 2012. Turk Bootstrap Word Sense Inventory 2.0: A Large-Scale Resource for Lexical Substitution. In *Proceedings of the 8th International Conference on Language Resources and Evaluation*, pages 4038–4042, Istanbul, Turkey. European Language Resources Association.
- José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. NASARI: a novel approach to a semantically-aware representation of items. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 567–577, Denver, CO, USA. Association for Computational Linguistics.
- Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. 2014. A unified model for word sense representation and disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1025–1035, Doha, Qatar. Association for Computational Linguistics.
- Marie-Catherine De Marneffe, Bill MacCartney, Christopher D. Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th Language Resources and Evaluation Conference (LREC'2006)*, pages 449–454, Genova, Italy. European Language Resources Association.
- Antonio Di Marco and Roberto Navigli. 2013. Clustering and diversifying web search results with graph-based word sense induction. *Computational Linguistics*, 39(3):709–754.
- Beate Dorow, Dominic Widdows, Katarina Ling, Jean-Pierre Eckmann, Danilo Sergi, and Elisha Moses. 2005. Using curvature and markov clustering in graphs for lexical acquisition and word sense discrimination. In *Proceedings of the Meaning-2005 Workshop*, Trento, Italy.
- Ted Dunning. 1993. Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics*, 19:61–74.
- Martin Everett and Stephen P. Borgatti. 2005. Ego network betweenness. *Social networks*, 27(1):31–38.
- Stefan Evert. 2005. *The Statistics of Word Cooccurrences Word Pairs and Collocations*. Ph.D. thesis, University of Stuttgart.
- Stefano Faralli and Roberto Navigli. 2012. A new minimally-supervised framework for domain word sense disambiguation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1411–1422, Jeju Island, Korea, July. Association for Computational Linguistics.
- Alex A. Freitas. 2014. Comprehensible classification models: a position paper. *ACM SIGKDD Explorations Newsletter*, 15(1):1–10.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics.
- David Hope and Bill Keller. 2013. MaxMax: A Graph-based Soft Clustering Algorithm Applied to Word Sense Induction. In *Proceedings of the 14th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 368–381, Samos, Greece. Springer.

- Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL'2012)*, pages 873–882, Jeju Island, Korea. Association for Computational Linguistics.
- Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. SensEmbed: learning sense embeddings for word and relational similarity. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL'2015)*, pages 95–105, Beijing, China. Association for Computational Linguistics.
- Nancy Ide and Jean Véronis. 1998. Introduction to the special issue on word sense disambiguation: the state of the art. *Computational linguistics*, 24(1):2–40.
- David Jurgens and Ioannis Klapafitis. 2013. Semeval-2013 Task 13: Word Sense Induction for Graded and Non-graded Senses. In *Proceedings of the 2nd Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval'2013)*, pages 290–299, Montreal, Canada. Association for Computational Linguistics.
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan. Association for Computational Linguistics.
- Dan Klein, Kristina Toutanova, H. Tolga Ilhan, Sepandar D. Kamvar, and Christopher D. Manning. 2002. Combining Heterogeneous Classifiers for Word-Sense Disambiguation. In *Proceedings of the ACL'2002 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, volume 8, pages 74–80, Philadelphia, PA, USA. Association for Computational Linguistics.
- Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP-95)*, volume 1, pages 181–184, Detroit, MI, USA. IEEE.
- Jey Han Lau, Paul Cook, and Timothy Baldwin. 2013. unimelb: Topic Modelling-based Word Sense Induction. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 307–311, Atlanta, GA, USA. Association for Computational Linguistics.
- Yoong Keok Lee and Hwee Tou Ng. 2002. An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'2002)*, volume 10, pages 41–48, Philadelphia, PA, USA. Association for Computational Linguistics.
- Michael Lesk. 1986. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In *Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26, Toronto, ON, Canada. ACM.
- Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, MD, USA. Association for Computational Linguistics.
- Jiwei Li and Dan Jurafsky. 2015. Do multi-sense embeddings improve natural language understanding? In *Conference on Empirical Methods in Natural Language Processing (EMNLP'2015)*, pages 1722–1732, Lisboa, Portugal. Association for Computational Linguistics.
- Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in nlp. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691, San Diego, CA, USA. Association for Computational Linguistics.
- Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning (ICML'1998)*, volume 98, pages 296–304, Madison, WI, USA. Morgan Kaufmann Publishers Inc.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Workshop at International Conference on Learning Representations (ICLR)*, pages 1310–1318, Scottsdale, AZ, USA.
- Tristan Miller, Chris Biemann, Torsten Zesch, and Iryna Gurevych. 2012. Using distributional similarity for lexical expansion in knowledge-based word sense disambiguation. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 1781–1796, Mumbai, India. Association for Computational Linguistics.
- George A. Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.
- Andrea Moro and Roberto Navigli. 2015. SemEval-2015 Task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 288–297, Denver, CO, USA. Association for Computational Linguistics.

- Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244.
- Roberto Navigli and Simone Paolo Ponzetto. 2010. Babelnet: Building a very large multilingual semantic network. In *Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics*, pages 216–225, Uppsala, Sweden.
- Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10.
- Arvind Neelakantan, Jeevan Shankar, Alexandre Pasos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069, Doha, Qatar. Association for Computational Linguistics.
- Hwee Tou Ng. 1997. Exemplar-Based Word Sense Disambiguation: Some Recent Improvements. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 208–213, Providence, RI, USA. Association for Computational Linguistics.
- Luis Nieto Piña and Richard Johansson. 2016. Embedding senses for efficient graph-based word sense disambiguation. In *Proceedings of TextGraphs-10: the Workshop on Graph-based Methods for Natural Language Processing*, pages 1–5, San Diego, CA, USA. Association for Computational Linguistics.
- Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.
- Alexander Panchenko and Olga Morozova. 2012. A study of hybrid similarity measures for semantic relation extraction. In *Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data*, pages 10–18, Avignon, France. Association for Computational Linguistics.
- Alexander Panchenko. 2016. Best of both worlds: Making word sense embeddings interpretable. In *Proceedings of the 10th Language Resources and Evaluation Conference (LREC'2016)*, pages 2649–2655, Portorož, Slovenia. European Language Resources Association (ELRA).
- Patrick Pantel and Dekang Lin. 2002. Discovering word senses from text. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 613–619, Edmonton, AB, Canada.
- Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. 2016. Attentive explanations: Justifying decisions and pointing to the evidence. *arXiv preprint arXiv:1612.04757*.
- Ted Pedersen and Rebecca Bruce. 1997. Distinguishing word senses in untagged text. In *Proceedings of the 2nd Conference on Empirical Methods in Natural Language Processing (EMNLP'1997)*, pages 197–207, Providence, RI, USA. Association for Computational Linguistics.
- Ted Pedersen, Satanjeev Banerjee, and Siddharth Patwardhan. 2005. Maximizing semantic relatedness to perform word sense disambiguation. *University of Minnesota supercomputing institute research report UMSI*, 25:2005.
- Maria Pelevina, Nikolay Arefiev, Chris Biemann, and Alexander Panchenko. 2016. Making sense of word embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 174–183, Berlin, Germany. Association for Computational Linguistics.
- Martin Riedl. 2016. *Unsupervised Methods for Learning and Using Semantics of Natural Language*. Ph.D. thesis, Technische Universität Darmstadt, Darmstadt.
- Sascha Rothe and Hinrich Schütze. 2015. Autoextend: Extending word embeddings to embeddings for synsets and lexemes. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1793–1803, Beijing, China. Association for Computational Linguistics.
- Eugen Ruppert, Manuel Kaufmann, Martin Riedl, and Chris Biemann. 2015. Jobimviz: A web-based visualization for graph-based distributional semantic models. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 103–108, Beijing, China. Association for Computational Linguistics and The Asian Federation of Natural Language Processing.
- Hinrich Schütze. 1998. Automatic Word Sense Discrimination. *Computational Linguistics*, 24(1):97–123.
- Fei Tian, Hanjun Dai, Jiang Bian, Bin Gao, Rui Zhang, Enhong Chen, and Tie-Yan Liu. 2014. A probabilistic model for learning multi-prototype word embeddings. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING'2014)*, pages 151–160, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.
- Tim Van de Cruys. 2010. Mining for meaning: The extraction of lexicosemantic knowledge from text. *Groningen Dissertations in Linguistics*, 82.
- Stijn Van Dongen. 2008. Graph clustering via a discrete uncoupling process. *SIAM Journal on Matrix Analysis and Applications*, 30(1):121–141.

Alfredo Vellido, José David Martín, Fabrice Rossi, and Paulo J.G. Lisboa. 2011. Seeing is believing: The importance of visualization in real-world machine learning applications. In *Proceedings of the 19th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN'2011)*, pages 219–226, Bruges, Belgium.

Alfredo Vellido, José D. Martín-Guerrero, and Paulo J.G. Lisboa. 2012. Making machine learning models interpretable. In *20th European Symposium on Artificial Neural Networks, ESANN*, volume 12, pages 163–172, Bruges, Belgium.

Jean Véronis. 2004. HyperLex: Lexical cartography for information retrieval. *Computer Speech and Language*, 18:223–252.

Heng Low Wee. 2010. Word Sense Prediction Using Decision Trees. Technical report, Department of Computer Science, National University of Singapore.

Dominic Widdows and Beate Dorow. 2002. A graph model for unsupervised lexical acquisition. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING'2002)*, pages 1–7, Taipei, Taiwan. Association for Computational Linguistics.

Deniz Yuret. 2012. FASTSUBS: An efficient and exact procedure for finding the most likely lexical substitutes based on an n-gram language model. *IEEE Signal Processing Letters*, 19(11):725–728.

Zhi Zhong and Hwee Tou Ng. 2010. It makes sense: A wide-coverage word sense disambiguation system for free text. In *Proceedings of the ACL 2010 System Demonstrations*, pages 78–83, Uppsala, Sweden. Association for Computational Linguistics.

A.2 Paper “Always Keep your Target in Mind: Studying Semantics and Improving Performance of Neural Lexical Substitution”

N. Arefyev, B. Sheludko, A. Podolskiy, and A. Panchenko, “Always Keep your Target in Mind: Studying Semantics and Improving Performance of Neural Lexical Substitution,” in *Proceedings of the 28th International Conference on Computational Linguistics*, (Barcelona, Spain (Online)), pp. 1242–1255, International Committee on Computational Linguistics, Dec. 2020

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/2020.coling-main.107>

Always Keep your Target in Mind: Studying Semantics and Improving Performance of Neural Lexical Substitution

Nikolay Arefyev^{1, 2, 3}, Boris Sheludko^{1, 2}, Alexander Podolskiy^{1*}, and Alexander Panchenko⁴

¹Samsung Research Center Russia, Moscow, Russia

²Lomonosov Moscow State University, Moscow, Russia

³HSE University, Moscow, Russia

⁴Skolkovo Institute of Science and Technology, Moscow, Russia

narefjev@cs.msu.ru

{b.sheludko, a.podolskiy}@samsung.com

a.panchenko@skoltech.ru

Abstract

Lexical substitution, i.e. generation of plausible words that can replace a particular target word in a given context, is an extremely powerful technology that can be used as a backbone of various NLP applications, including word sense induction and disambiguation, lexical relation extraction, data augmentation, etc. In this paper, we present a large-scale comparative study of lexical substitution methods employing both rather old and most recent language and masked language models (LMs and MLMs), such as context2vec, ELMo, BERT, RoBERTa, XLNet. We show that already competitive results achieved by SOTA LMs/MLMs can be further substantially improved if information about the target word is injected properly. Several existing and new target word injection methods are compared for each LM/MLM using both intrinsic evaluation on lexical substitution datasets and extrinsic evaluation on word sense induction (WSI) datasets. On two WSI datasets we obtain new SOTA results. Besides, we analyze the types of semantic relations between target words and their substitutes generated by different models or given by annotators.

1 Introduction

Lexical substitution is the task of generating words that can replace a given word in a given textual context. For instance, in the sentence “*My daughter purchased a new car*” the word *car* can be substituted by its synonym *automobile*, but also with co-hyponym *bike*, or even hypernym *motor vehicle* while keeping the original sentence grammatical. Lexical substitution has been proven effective in various applications, such as word sense induction (Amrami and Goldberg, 2018), lexical relation extraction (Schick and Schütze, 2020), paraphrase generation, text simplification, textual data augmentation, etc. Note that the preferable type (e.g., synonym, hypernym, co-hyponym, etc.) of generated substitutes depends on the task at hand.

The new generation of language and masked language models (LMs/MLMs) based on deep neural networks enabled a profound breakthrough in almost all NLP tasks. These models are commonly used to perform pre-training of deep neural networks, which are then fine-tuned to some final task different from language modeling. However, in this paper we study how the progress in unsupervised pre-training over the last five years affected the quality of lexical substitution. We adapt context2vec (Melamud et al., 2016), ELMo (Peters et al., 2018), BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and XLNet (Yang et al., 2019) to solve lexical substitution task without any fine-tuning, but using additional techniques to ensure similarity of substitutes to the target word, which we call target word injection techniques. We provide the first large-scale comparison of various neural LMs/MLMs with several target word injection methods on lexical substitution and WSI tasks. Our research questions are the following (i) which pre-trained models are the best for substitution in context, (ii) additionally to pre-training larger

*Left Samsung

This work is licensed under a Creative Commons Attribution 4.0 International Licence. Licence details: <http://creativecommons.org/licenses/by/4.0/>.

models on more data, what are the other ways to improve lexical substitution, and (iii) what are the generated substitutes semantically. More specifically, the main contributions of the paper are as follows¹:

- A comparative study of five neural LMs/MLMs applied for lexical substitution based on both intrinsic and extrinsic evaluation.
- A study of methods of target word injection for further lexical substitution quality improvement.
- An analysis of types of semantic relations (synonyms, hypernyms, co-hyponyms, etc.) produced by neural substitution models as well as human annotators.

2 Related Work

Solving the lexical substitution task requires finding words that are both appropriate in the given context and related to the target word in some sense (which may vary depending on the application of generated substitutes). To achieve this, unsupervised substitution models heavily rely on distributional similarity models of words (DSMs) and language models (LMs). Probably, the most commonly used DSM is *word2vec* model (Mikolov et al., 2013). It learns word embeddings and context embeddings to be similar when they tend to occur together, resulting in similar embeddings for distributionally similar words. Contexts are either nearby words or syntactically related words (Levy and Goldberg, 2014). In (Melamud et al., 2015b) several metrics for lexical substitution were proposed based on embedding similarity of substitutes both to the target word and to the words in the given context. Later (Roller and Erk, 2016) improved this approach by switching to dot-product instead of cosine similarity and applying an additional trainable transformation to context word embeddings.

A more sophisticated *context2vec* model producing embeddings for a word in a particular context (contextualized word embeddings) was proposed in (Melamud et al., 2016) and was shown to outperform previous models in a ranking scenario when candidate substitutes are given. The training objective is similar to *word2vec*, but context representation is produced by two LSTMs (a forward and a backward for the left and the right context), in which final outputs are combined by feed-forward layers. For lexical substitution, candidate word embeddings are ranked by their similarity to the given context representation. A similar architecture consisting of a forward and a backward LSTM is employed in ELMo (Peters et al., 2018). However, in ELMo each LSTM was trained with the LM objective instead. To rank candidate substitutes using ELMo (Soler et al., 2019) proposed calculating cosine similarity between contextualized ELMo embeddings of the target word and all candidate substitutes. This requires feeding the original example with the target word replaced by one of the candidate substitutes at a time. The average of outputs at the target timestep from all ELMo layers performed best. However, they found *context2vec* performing even better and explained this by the negative sampling training objective, which is more related to the task.

Recently, Transformer-based models pre-trained on huge corpora with LM or similar objectives have shown SOTA results in various NLP tasks. BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) were trained to restore a word replaced with a special [MASK] token given its full left and right contexts (masked LM objective), while XLNet (Yang et al., 2019) predicted a word at a specified position given only some randomly selected words from its context (permutation LM objective). In (Zhou et al., 2019), BERT was reported to perform poorly for lexical substitution (which is contrary to our experiments), and two improvements were proposed to achieve SOTA results using it. Firstly, dropout is applied to the target word embedding before showing it to the model. Secondly, the similarity between the original contextualized representations of the context words and their representations after replacing the target by one of the possible substitutes are integrated into the ranking metric to ensure minimal changes in the sentence meaning. This approach is very computationally expensive, requiring calculation of several forward passes of BERT for each input example, depending on the number of possible substitutes. We are not aware of any work applying XLNet for lexical substitution, but our experiments show that it outperforms BERT by a large margin.

¹The repository for this paper: <https://github.com/bsheludko/lexical-substitution>

Supervised approaches to lexical substitution include (Szarvas et al., 2013a; Szarvas et al., 2013b; Hintz and Biemann, 2016). These approaches rely on manually curated lexical resources like WordNet, so they are not easily transferable to different languages, unlike those described above. Also, the latest unsupervised methods were shown to perform better (Zhou et al., 2019).

3 Neural Language Models for Lexical Substitution with Target Word Injection

To generate substitutes, we introduce several substitute probability estimators, which are models taking a text fragment and a target word position in it as input and producing a list of substitutes with their probabilities. To build our substitute probability estimators we employ the following LMs/MLMs: context2vec (Melamud et al., 2016), ELMo (Peters et al., 2018), BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and XLNet (Yang et al., 2019). These models were selected to represent the progress in unsupervised pre-training with language modeling and similar tasks over the last five years. Given a target word occurrence, the basic approach for models like context2vec and ELMo is to encode its context and predict the probability distribution over possible center words in this particular context. This way, the model does not see the target word. For MLMs, the same result can be achieved by masking the target word. This basic approach employs the core ability of LMs/MLMs of predicting words that fit a particular context. However, these words are often not related to the target. The information about the target word can improve generated substitutes, but what is the best method of injecting this information is an open question.

3.1 Target Word Injection Methods

We experiment with several methods to introduce information about the original target word into neural lexical substitution models and show that their performance differs significantly. Suppose we have an example LTR , where T is the target word, and $C = (L, R)$ is its context (left and right correspondingly). For instance, the occurrence of the target word *fly* in the sentence “*Let me fly away!*” will be represented as $T = “fly”$, $L = “Let me”$, $R = “away!”$.

+embs This method combines a distribution provided by a context-based substitute probability estimator $P(s|C)$ with a distribution based on the proximity of possible substitutes to the target $P(s|T)$. The proximity is computed as the inner product between the respective embeddings, and the softmax function is applied to get a probability distribution. However, if we simply multiply these distributions, the second will have almost no effect because the first is very peaky. To align the orders of distributions, we use temperature softmax with carefully selected temperature hyperparameter: $P(s|T) \propto \exp(\frac{\langle emb_s, emb_T \rangle}{\tau})$. The final distribution is obtained by the formula $P(s|C, T) \propto \frac{P(s|C)P(s|T)}{P(s)^\beta}$. For $\beta = 1$, this formula can be derived by applying the Bayes rule and assuming conditional independence of C and T given s . Other values of β can be used to penalize frequent words, more or less. Our current methods are limited to generating only substitutes from the vocabulary of the underlying LM/MLM. Thus, we take word or subword embeddings of the same model we apply the injection to. Other word embeddings like word2vec may perform better, but we leave these experiments for future work.

Word probabilities $P(s)$ are retrieved from wordfreq library² for all models except ELMo. Following (Arefyev et al., 2019), for ELMo, we calculate word probabilities from word ranks in the ELMo vocabulary (which is ordered by word frequencies) based on Zipf-Mandelbrot distribution and found it performing better presumably due to better correspondence to the corpus ELMo was pre-trained on.

Dynamic patterns Following the approach proposed in (Amrami and Goldberg, 2018), we replace the target word T by “ T and $_$ ” (e.g. “*Let me fly and _ away!*”). Then some LM/MLM is employed to predict possible words at timestep “ $_$ ”. Thus, dynamic patterns provide information about the target word to the model via Hearst-like patterns.

Duplicate input This method duplicates the original example while hiding the target word (e.g., “*Let me fly away! Let me _ away!*”). Then possible words at timestep “ $_$ ” are predicted. It is based on our

²<https://pypi.org/project/wordfreq>

observation that Transformer-based MLMs are very good at predicting words from the context when they fit the specified timestep (copying) while still giving a high probability to their distributionally similar alternatives.

Original input For MLMs such as BERT and RoBERTa, instead of masking the target word, we can leave it intact. Thus, the model predicts possible words at the target position while receiving the original target at its input. We noticed that unlike duplicate input, in this case, the MLM often puts the whole probability mass to the original target and gives very small probabilities to other words making their ranking less reliable. For XLNet, we can use such attention masks that the context words can see the target word in the content stream. Thus, the content stream becomes a full self-attention layer and sees all words in the original example. We do not apply the original input technique with context2vec and ELMo since, for these models, there is no reasonable representation that can be used to predict possible words at some timestep while depending on the input at that timestep, at least without fine-tuning. For other models, this option significantly outperforms target word masking and does not require many additional efforts. Hence, if not specified otherwise, we use it in our experiments with pure BERT, RoBERTa, XLNet estimators, and in +embs method when estimating $P(s|C)$ with BERT and XLNet.

3.2 Neural Language Models for Lexical Substitution

Different LMs/MLMs are employed as described below to obtain context-based substitute probability distribution $P(s|C)$. For each of them, we experiment with different target injection methods.

C2V Context2vec encodes left, and right context separately using its forward and backward LSTM layers correspondingly and then combines their outputs with two feed-forward layers producing the final full context representation. Possible substitutes are ranked by the dot product similarity of their embeddings and the context representation. We use the original implementation³ and the weights⁴ pre-trained on ukWac dataset.

ELMo To encode left and right context with ELMo, we use its forward and backward stacked LSTMs, which were pre-trained with LM objective. However, there are no pre-trained layers to combine their outputs. Thus, we obtain two independent distributions over possible substitutes: one given the left context $P(s|L)$, another given the right context $P(s|R)$. To combine these distributions we use BComb-LMs method proposed in (Arefyev et al., 2019), which can be derived similarly to +embs method described above: $P(s|L, R) = \frac{P(s|L)P(s|R)}{P^{\gamma}(s)}$. The original version of ELMo described in (Peters et al., 2018) is used, which is the largest version pre-trained on 1B Word Corpus.

BERT/RoBERTa By default, we give our full example without any masking as input and calculate the distribution over the model’s vocabulary at the position of the first subword of the target word. We employ BERT-large-cased and RoBERTa-large models as the best-performing ones. Unlike BERT and XLNet, we found that RoBERTa with +embs injection method performs better if cosine similarity instead of dot-product similarity is used when estimating $P(s|T)$ and the target word is masked when estimating $P(s|C)$. Thus, in the following experiments, we use these choices by default for RoBERTa+embs model.

XLNet By default, we use the XLNet-large-cased model with the original input, obtaining substitute probability distribution similarly to BERT. We found that for short contexts, XLNet performance degrades. To mitigate this problem, we prepend the initial context with some text ending with the end-of-document special token.

4 Baseline Lexical Substitution Models

Lexical substitution models described above are compared to the best previously published results, as well as our re-implementations of the following baseline models proposed in (Roller and Erk, 2016).

³<https://github.com/orenmel/context2vec>

⁴<https://u.cs.biu.ac.il/~nlp/resources/downloads/context2vec/>

OOC Out of Context model ranks possible substitutes by their cosine similarity to the target word and completely ignores given context. Following (Roller and Erk, 2016) we use dependency-based embeddings⁵ released by (Melamud et al., 2015b).

nPIC Non-parameterized Probability In Context model returns the product of two distributions measuring the fitness of a substitute to the context and to the target: $nPIC(s|T, C) = P(s|T) \times P_n(s|C)$, where $P(s|T) \propto \exp(\langle embs_s, embst \rangle)$ and $P_n(s|C) \propto \exp(\sum_{c \in C} \langle embs_s, embcs' \rangle)$. Here $embs$ and $embs'$ are dependency-based word and context embeddings, and C are those words that are directly connected to the target in the dependency tree.

5 Intrinsic Evaluation

We perform an intrinsic evaluation of the proposed models on two lexical substitution datasets.

5.1 Experimental Setup

Lexical substitution task is concerned with finding appropriate substitutes for a target word in a given context. This task was originally introduced in SemEval 2007 Task 10 (McCarthy and Navigli, 2007) to evaluate how distributional models handle polysemous words. In the lexical substitution task, annotators are provided with a target word and its context. Their task is to propose possible substitutes. Since there are several annotators, we have some weight for each possible substitute in each example, which is equal to the number of annotators provided this substitute.

We rank substitutes for a target word in a context by acquiring probability distribution over vocabulary on the target position. Lexical substitution task comes with two variations: candidate ranking and all-words ranking. In candidate ranking task, models are provided with a list of candidates. Following previous works, we acquire this list by merging all gold substitutes of the target lemma over the corpus. We measure performance on this task with Generalized Average Precision (GAP) that was introduced in (Thater et al., 2010). GAP is similar to Mean Average Precision, and the difference is in the weights of substitutes: the higher the weight of the word, the higher it should be ranked. Following (Melamud et al., 2015a), we discard all multi-word expressions from the gold substitutes and omit all instances that are left without gold substitutes.

In the all-vocab ranking task, models are not provided with candidate substitutes. Therefore, it is much harder task than the previous one. Models shall give higher probabilities to gold substitutes than to all other words in their vocabulary usually containing hundreds of thousands of words. Following (Roller and Erk, 2016), we calculate the precision of the top 1 and 3 predictions ($P@1$, $P@3$) as an evaluation metric for the all-ranking task. Additionally, we look at the recall of top 10 predictions ($R@10$).

The following lexical substitution datasets are used:

SemEval 2007 Task 10 (McCarthy and Navigli, 2007) consists of 2010 sentences for 201 polysemous words, 10 sentences for each. Annotators were asked to give up to 3 possible substitutes.

CoInCo or Concepts-In-Context dataset (Kremer et al., 2014) consists of about 2500 sentences that come from fiction and news. In these sentences, each content word is a separate example, resulting in about 15K examples. Annotators provided at least 6 substitutes for each example.

5.2 Results and Discussion

Comparison to previously published results Table 1 contains metrics for candidate and all-vocab ranking tasks. We compare our best model (XLNet+embs) with the best previously published results presented in (Roller and Erk, 2016), context2vec (c2v) model (Melamud et al., 2016) and BERT for lexical substitution presented in (Zhou et al., 2019). The proposed model outperforms solid models such as PIC, c2v, and substitute vector by a large margin on both ranking tasks. Nevertheless, (Zhou et al., 2019) reported better results than XLNet+embs in both lexical substitution tasks. In (Zhou et al., 2019), authors add a substitute validation metric that measures the fitness of a substitute to a context. It is

⁵http://www.cs.biu.ac.il/~nlp/resources/downloads/lexsub_embeddings

⁶We could not reproduce the results of (Zhou et al., 2019) and their code was not available.

Model	SemEval 2007			CoInCo		
	GAP	P@1	P@3	GAP	P@1	P@3
Transfer Learning (Hintz and Biemann, 2016)	51.9	-	-	-	-	-
PIC (Roller and Erk, 2016)	52.4	19.7	14.8	48.3	18.2	13.8
Supervised Learning (Szarvas et al., 2013b)	55.0	-	-	-	-	-
Substitute vector (Melamud et al., 2015a)	55.1	-	-	50.2	-	-
context2vec (Melamud et al., 2016)	56.0	-	-	47.9	-	-
BERT for lexical substitution (Zhou et al., 2019) ⁶	60.5	51.1	-	57.6	56.3	-
XLNet+embs	59.6	49.5	34.9	55.6	51.5	39.9
XLNet+embs (w/o target exclusion)	59.6	0.4	26.0	53.5	2.5	30.0
XLNet+embs (w/o lemmatization)	59.2	38.3	27.5	53.2	34.5	27.1
XLNet+embs ($\mathcal{T} = 1.0$)	52.6	34.8	24.6	49.4	40.5	30.4

Table 1: Intrinsic evaluation of our best model and its variations on lexical substitution datasets.

computed as the weighted sum of cosine similarities of contextualized representations of words in two sentence versions: original and one where the target word is replaced with the substitute. This technique substantially improves predictions. However, substitute validation requires additional forward passes, hence, increasing computational overhead. Our methods need only one forward pass. Our approach is orthogonal to the substitute validation. Thus, a combination of two methods can improve results further. It is worth mentioning that BERT and XLNet work on a subword level. Hence, their vocabularies are much smaller in size (30K subwords) than those of ELMo (800K words) or C2V (180K words) and contain only a fraction of possible substitutes. Thus, these models can be significantly improved by generating multi-token substitutes.

Additionally, table 1 includes ablation analysis of our best model. Using ordinary softmax (which is equivalent to setting $\mathcal{T} = 1.0$) results in all metrics decreasing by a large margin. Also, post-processing of substitutes has a significant impact on all-words ranking metrics. Since LMs/MLMs generate grammatically plausible word forms and often generate the target word itself among top substitutes, additional lemmatization and target exclusion is required to match gold substitutes. In (Roller and Erk, 2016), the authors used the NLTK English stemmer to exclude all forms of the target word. In (Melamud et al., 2016) NLTK WordNet lemmatizer is used to lemmatize only candidates. For a fair comparison, the same post-processing is used for all models in the following experiments.

Model	SemEval 2007				CoInCo			
	GAP	P@1	P@3	R@10	GAP	P@1	P@3	R@10
OOC	44.65	16.82	12.83	18.36	46.31	19.58	15.03	12.99
nPIC	52.46	23.22	17.61	27.4	48.57	25.75	19.12	17.33
C2V	55.81	7.79	5.92	11.03	48.32	8.01	6.63	7.54
C2V+embs	53.40	28.01	21.72	33.02	50.73	29.64	24.0	21.37
ELMo	53.63	11.73	8.59	13.93	49.47	13.66	10.87	11.34
ELMo+embs	54.15	31.95	22.20	31.82	52.22	35.92	26.6	23.80
BERT	54.40	38.34	27.71	39.72	50.50	42.56	32.64	28.63
BERT+embs	53.88	41.64	30.57	43.48	50.85	46.05	35.63	31.37
RoBERTa	56.73	32.00	24.35	36.89	50.63	34.77	27.15	25.12
RoBERTa+embs	58.83	44.13	31.67	44.70	54.68	46.54	36.33	32.03
XLNet	59.10	31.70	22.80	34.90	53.39	38.16	28.58	26.46
XLNet+embs	59.62	49.48	34.88	47.18	55.63	51.48	39.91	34.91

Table 2: Comparison of our models and re-implemented baselines with the same post-processing.

Re-implementation of the baselines In Table 2, we compare our models based on different LMs/MLMs with and without +embs injection technique. Remember that BERT, RoBERTa, and XLNet see the target even if +embs is not applied, thus, providing already strong baseline results. All compared models, including re-implemented OOC and nPIC, employ the same post-processing consisting of substitute lemmatization followed by target exclusion. First, we notice that our best substitution models substantially outperform word2vec based PIC and OOC methods. For example, the XLNet+embs gives 2x better P@1 and P@3 than the baselines. This indicates that proposed models are better at capturing the meaning of a

word in a context as such, providing more accurate substitutes. On the candidate ranking task bare C2V model outperforms ELMo and BERT based models, but it shows the lowest Precision@1. We note that +embs technique substantially improves the performance of all models in all-vocab ranking task, and also increases GAP for the majority of models.

Injection of information about target word Next, we compare target injection methods described in Section 3. Figure 1 presents the Recall@10 metric for all of our neural substitution models with each applicable target injection method.

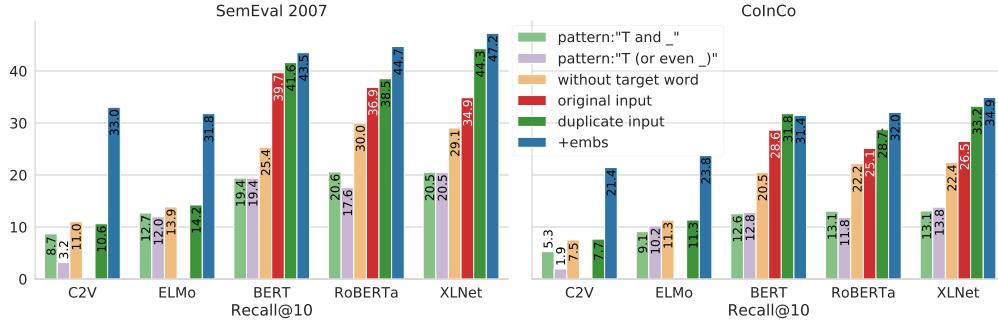


Figure 1: Comparison of various target information injection methods (SemEval 2007 and CoInCo datasets).

Application of dynamic patterns leads to lower performance even compared to the models that do not see the target word. Although we show the target word to the substitute generator, the pattern can spoil predictions, e.g., using “T and _” pattern with a verb can produce words denoting subsequent actions related to the verb, but not its synonyms. When we use the original input without any masking, the models produce substitutes more related to the target, resulting in good baseline performance. Applying the +embs method leads to a significant increase of Recall@10 for all models, almost always being the best performing injection method. For C2V and ELMo, it gives 2-3 times better recall than all other injection methods. Duplicating input performs surprisingly good for Transformer-based models but does not help for LSTM-based C2V and ELMo. This is likely related to the previous observations that Transformers are very good at copying words from the surrounding context, thus, predicting the original target word, but also words with similar embeddings. The highest impact is for the XLNet model as it can not straightforwardly use information from the target position due to its autoregressive nature but can easily find the target in the sentence copy. Overall, our experiments show that proper injection of the information about the target word results in much more plausible substitutes generated.

Hyperparameters Since there is no development set of reasonable size in SemEval 2007 dataset, we decided to select hyperparameters for SemEval 2007 on CoInCo and vice versa. However, the selected hyperparameters turned out to be the same. Thus, for both datasets we use $\mathcal{T}=0.1$ for BERT+embs, XLNet+embs and ELMo+embs models, $\mathcal{T}=0.25$ for RoBERTa+embs and $\mathcal{T}=1.0$ for C2V+embs. For BERT+embs, XLNet+embs, RoBerta+embs, C2V+embs $\beta=0.0$. For ELMo and ELMo+embs $\gamma=0.5$, $\beta=1.5$.

6 Extrinsic Evaluation

In this section, we perform the extrinsic evaluation of our models applied to the Word Sense Induction (WSI) task. The data for this task commonly consists of a list of ambiguous target words and a corpus of sentences containing these words. Models are required to cluster all occurrences of each target word according to their meaning. Thus, the senses of all target words are discovered in an unsupervised fashion. For example, suppose that we have the following sentences with the target word *bank*:

1. *He settled down on the river bank and contemplated the beauty of nature,*

2. *They unloaded the tackle from the boat to the bank.*
3. *Grand River bank now offers a profitable mortgage.*

Sentences 1 and 2 shall be put in one cluster, while sentence 3 must be assigned to another. This task was proposed in several SemEval competitions (Agirre and Soroa, 2007; Manandhar et al., 2010; Jurgens and Klapaftis, 2013). The current state-of-the-art approach (Amrami and Goldberg, 2019) rely on substitute vectors, i.e., each word usage is represented as a substitute vector based on the most probable substitutes, then clustering is performed over these substitute vectors.

Model	SemEval-2010 (AVG)	SemEval-2013 (AVG)
(Amrami and Goldberg, 2018)	–	25.43±0.48
(Amrami and Goldberg, 2019)	53.6±1.2	37.0±0.5
C2V	38.9	18.2
C2V+embs	28.5	21.7
ELMo	41.8	27.6
ELMo+embs	45.3	28.2
BERT	52.0	34.5
BERT+embs	53.8	36.8
RoBERTa	49.6	34.0
RoBERTa+embs	51.4	34.5
XLNet	52.2	33.4
XLNet+embs	54.2	37.3

Table 3: Extrinsic evaluation on word sense induction datasets.

We implemented a WSI algorithm using lexical substitutes from our models. The algorithm is a simplified version of methods described in (Amrami and Goldberg, 2019; Arefyev et al., 2019). In the first step, we generate substitutes for each example, lemmatize them and take 200 most probable ones. We treat these 200 substitutes as a document. Then, TF-IDF vectors for these documents are calculated and clustered using agglomerative clustering with average linkage and cosine distance. The number of clusters maximizing the silhouette score is selected for each word individually.

In table 3 we compare our lexical substitution models on two WSI datasets. For the previous SOTA models, which are stochastic algorithms, the mean and the standard deviation are reported. Our WSI algorithm is deterministic; hence, we report the results of a single run. Our best model achieves higher metrics than the previous SOTA on both datasets, however, the difference is within one standard deviation. Similarly to the intrinsic evaluation results, our +embs injection method substantially improves the performance of all models for WSI, except for the C2V+embs model on SemEval-2010, which probably used suboptimal hyperparameters.

Hyperparameters The optimal hyperparameters for WSI models were selected on the TWSI dataset (Biemann, 2012). For both evaluation datasets we used $\beta=2.0$ for BERT+embs, XLNet+embs, RoBERTa+embs and ELMo+embs models, $\beta=0.0$ for C2V+embs, $\mathcal{T}=2.5$ for BERT+embs, $\mathcal{T}=1.0$ for XLNet+embs, $\mathcal{T}=10.0$ for RoBERTa+embs, $\mathcal{T}=0.385$ for ELMo+embs and $\mathcal{T}=1.0$ for C2V+embs.

7 Analysis of Semantic Relation Types

In this section we analyze the types of substitutes produced by different neural substitution models.

7.1 Experimental Setup

For this analysis, the CoInCo lexical substitution dataset (Kremer et al., 2014) described above is used. We employ WordNet (Miller, 1995) to find the relationship between a target word and each generated substitute. First, from all possible WordNet synsets two synsets containing the target word and its substitute with the shortest path between them are selected. Then relation between these synsets is identified as follows. If there is a direct relation between the synsets, i.e. synonymy, hyponymy, hypernymy, or co-hyponymy with a common direct hypernym, we return this relation. Otherwise, we search for an indirect relation, i.e. transitive hyponymy or hypernymy, or co-hyponymy with a common hypernym at a distance of maximum 3 hops from each synset. We also introduce several auxiliary relations: *unknown-word* – the

target or the substitute is not found among WordNet synsets with the required PoS, *unknown-relation* – the target and the substitute are in the same WordNet tree, but no relation can be assigned among those described above, *no-path* – the target and the substitute are in different trees.

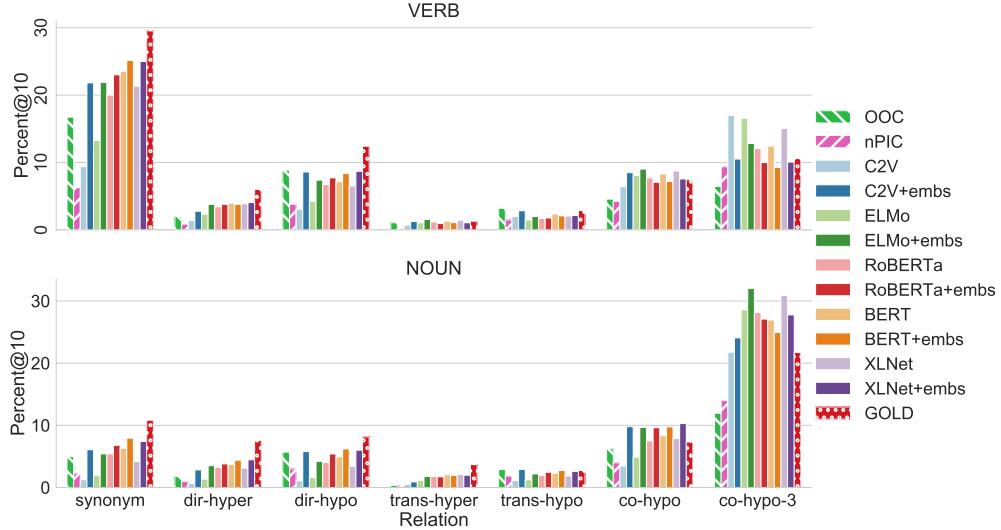


Figure 2: Proportions of substitutes related to the target by various semantic relations according to WordNet. We took top 10 substitutes from each model and all substitutes from the gold standard.

7.2 Discussion of Results

For nouns and verbs the proportions of non-auxiliary relations are shown in figure 2, for all words and relations see Appendix B. Our analysis shows that a substantial fraction of substitutes has no direct relation to the target word in terms of WordNet, even in case of the gold standard substitutes. Besides, even human annotators occasionally provide substitutes of incorrect PoS, e.g., for *bright* as an adjective there is the verb *glitter* among gold substitutes. For adjectives and adverbs 18% and 25% of gold substitutes are unknown words (absent among synsets with the correct PoS), while for verbs and nouns less than 7% are unknown. For baseline models OOC and nPIC, the overwhelming number of substitutes are unknown words. One of the reasons for this might be the fact that their vocabularies contain words with typos, but we also noticed that these models frequently do not preserve the PoS of the target word. The models based on LMs/MLMs produce much fewer unknown substitutes. Surprisingly, our +embs target injection method further reduces the number of such substitutes, achieving the proportion comparable to the gold standard. We can therefore suggest that our injection method helps to better preserve the correct PoS even for SOTA MLMs.

For both nouns and verbs, +embs target injection method consistently increases the proportions of synonyms, direct hyponyms and direct hypernyms, while decreasing the proportions of distantly related co-hyponyms (co-hypo-3) or unrelated substitutes. This is more similar to the proportions in human substitutes. Thus, the addition of information from embeddings forces the models to produce words that are more closely related to the target word and more similar to human answers. For C2V and ELMo, which have no information on the target word, target word injection results in 2x-3x more synonyms generated.

For several sentences from SemEval 2007 dataset (McCarthy and Navigli, 2007), Figure 3 shows some examples of substitutes provided by the human annotators (GOLD) and generated by several models, see Appendix A for more examples and models. The first example shows the case when +embs injection method improves the result, ranking closely related substitutes, such as *telephone*, *cellphone*, higher. The

We were not able to travel in the weather , and there was no phone .										
GOLD	telephone (5)									
OOC	phone	telephone	phones	cellphone	fone	videophone	handset	telephones	p990i	cell-phone
XLNet	electricity	internet	phone	power	telephone	car	water	communication	radio	tv
XLNet+embs	phone	telephone	phones	cellphone	internet	radio	electricity	iphone	car	computer
What happened to the big , new garbage can at Church and Chambers Streets ?										
GOLD	bin (4)	disposal (1)	container (1)							
OOC	can	could	should	would	will	must	might	to	may	ll
XLNet	can	dump	bin	truck	disposal	pit	heap	pile	container	stand
XLNet+embs	can	could	will	bin	cannot	dump	may	truck	disposal	stand
Types of semantic relations: synonym co-hyponym co-hyponym 3 target direct hypernym transitive hypernym direct hyponym transitive hyponym unknown-relation unknown-word										

Figure 3: Examples of top substitutes provided by annotators (GOLD), the baseline (OOC), and two presented models (XLNet and XLNet+embs). The target word in each sentence is in bold, true positives are in bold also. The weights of gold substitutes are given in brackets. Each substitute is colored according to its relation to the target word. Substitutes before post-processing are shown.

substitutes provided by the bare XLNet model, such as *electricity* and *internet*, could be used in this context, but all the annotators had preferred the synonym *telephone* instead. The second case is related to the failure of the proposed method. Bare XLNet model generated substitutes related to the correct sense of the ambiguous target word *can*, and has all three gold substitutes among its top 10 predictions. In contrast, XLNet+embs produced words that are related to the most frequent sense: *will*, *could*, *cannot*, etc. We hypothesize that this problem could potentially be alleviated by choosing individual temperature for each example based on the characteristics of the combined distributions; this is a possible direction for our further research.

8 Conclusion

We presented the first comparison of a wide range of LMs/MLMs with different target word injection methods on the tasks of lexical substitution and word sense induction. Our results are the following: (i) large pre-trained language models yield better results than previous unsupervised and supervised methods of lexical substitution; (ii) if properly done, the integration of information about the target word substantially improves the quality of lexical substitution models. The proposed target injection method based on a fusion of a context-based distribution $P(s|C)$ with a target similarity distribution $P(s|T)$ proved to be the best one. When applied to the XLNet model, it yields new SOTA results on two WSI datasets. Finally, we study the semantics of the produced substitutes. This information can be valuable for practitioners selecting the most appropriate lexical substitution method for a particular NLP application.

Acknowledgements

We thank the anonymous reviewers for the valuable feedback. The contribution of Nikolay Arefyev to the paper was partially done within the framework of the HSE University Basic Research Program funded by the Russian Academic Excellence Project “5-100”.

References

- Eneko Agirre and Aitor Soroa. 2007. SemEval-2007 task 02: Evaluating word sense induction and discrimination systems. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 7–12, Prague, Czech Republic, June. Association for Computational Linguistics.
- Asaf Amrami and Yoav Goldberg. 2018. Word sense induction with neural biLM and symmetric patterns. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4860–4867, Brussels, Belgium, October-November. Association for Computational Linguistics.
- Asaf Amrami and Yoav Goldberg. 2019. Towards better substitution-based word sense induction. *CoRR*, abs/1905.12598.

- Nikolay Arefyev, Boris Sheludko, and Alexander Panchenko. 2019. Combining lexical substitutes in neural word sense induction. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP'19)*, RANLP '19, pages 62–70, Varna, Bulgaria.
- Chris Biemann. 2012. Turk bootstrap word sense inventory 2.0: A large-scale resource for lexical substitution. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 4038–4042, Istanbul, Turkey, May. European Language Resources Association (ELRA).
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.
- Gerold Hintz and Chris Biemann. 2016. Language transfer learning for supervised lexical substitution. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 118–129, Berlin, Germany, August. Association for Computational Linguistics.
- David Jurgens and Ioannis Klapaftis. 2013. SemEval-2013 task 13: Word sense induction for graded and non-graded senses. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 290–299, Atlanta, Georgia, USA, June. Association for Computational Linguistics.
- Gerhard Kremer, Katrin Erk, Sebastian Padó, and Stefan Thater. 2014. What substitutes tell us - analysis of an “all-words” lexical substitution corpus. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 540–549, Gothenburg, Sweden, April. Association for Computational Linguistics.
- Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, Maryland, June. Association for Computational Linguistics.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.
- Suresh Manandhar, Ioannis Klapaftis, Dmitriy Dligach, and Sameer Pradhan. 2010. SemEval-2010 task 14: Word sense induction &disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 63–68, Uppsala, Sweden, July. Association for Computational Linguistics.
- Diana McCarthy and Roberto Navigli. 2007. SemEval-2007 task 10: English lexical substitution task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 48–53, Prague, Czech Republic, June. Association for Computational Linguistics.
- Oren Melamud, Ido Dagan, and Jacob Goldberger. 2015a. Modeling word meaning in context with substitute vectors. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 472–482, Denver, Colorado, May–June. Association for Computational Linguistics.
- Oren Melamud, Omer Levy, and Ido Dagan. 2015b. A simple word embedding model for lexical substitution. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 1–7, Denver, Colorado, June. Association for Computational Linguistics.
- Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional LSTM. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61, Berlin, Germany, August. Association for Computational Linguistics.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, august.
- George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.
- Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June. Association for Computational Linguistics.

- Stephen Roller and Katrin Erk. 2016. PIC a different word: A simple model for lexical substitution in context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1121–1126, San Diego, California, june. Association for Computational Linguistics.
- Timo Schick and Hinrich Schütze. 2020. Rare words: A major problem for contextualized embeddings and how to fix it by attentive mimicking. In *AAAI*, pages 8766–8774.
- Aina Gári Soler, Anne Cocos, Marianna Apidianaki, and Chris Callison-Burch. 2019. A comparison of context-sensitive models for lexical substitution. In *Proceedings of the 13th International Conference on Computational Semantics - Long Papers*, pages 271–282, Gothenburg, Sweden, 23–27 May. Association for Computational Linguistics.
- György Szarvas, Chris Biemann, and Iryna Gurevych. 2013a. Supervised all-words lexical substitution using delexicalized features. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1131–1141, Atlanta, Georgia, June. Association for Computational Linguistics.
- György Szarvas, Róbert Busa-Fekete, and Eyke Hüllermeier. 2013b. Learning to rank lexical substitutions. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1926–1932, Seattle, Washington, USA, October. Association for Computational Linguistics.
- Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2010. Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 948–957, Uppsala, Sweden, July. Association for Computational Linguistics.
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.
- Wangchunshu Zhou, Tao Ge, Ke Xu, Furu Wei, and Ming Zhou. 2019. BERT-based lexical substitution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3368–3373, Florence, Italy, July. Association for Computational Linguistics.

A Additional Examples of Lexical Substitutes

We were not able to travel in the weather , and there was no phone .												
GOLD	telephone (5)											
OOC	phone	telephone	phones	cellphone	fone	videophone	handset	telephones	p990i	cell-phone		
C2V	let-up	anti-climax	rain	wind	excuse	sunshine	fog	snow	gridlock	respite		
C2V+embs	phone	telephone	phones	cellphone	videophone	landline	voicemail	telephone/fax	telephones	email		
ELMo	electricity	danger	snow	indication	sign	reason	rain	escape	word	doubt		
ELMo+embs	phone	telephone	cellphone	phones	radio	electricity	e-mail	contact	computer	vehicle		
BERT	phone	telephone	phones	radio	connection	car	line	cable	backup	mobile		
BERT+embs	phone	telephone	phones	radio	cell	mobile	cable	car	call	connection		
ROBERTa	phone	money	power	tv	telephone	cellphone	internet	food	job	car		
ROBERTa+embs	phone	telephone	phones	cellphone	smartphone	internet	electricity	computer	radio	iphone		
XLNet	electricity	internet	phone	power	telephone	car	water	communication	radio	tv		
XLNet+embs	phone	telephone	phones	cellphone	internet	radio	electricity	iphone	car	computer		
What happened to the big , new garbage can at Church and Chambers Streets ?												
GOLD	bin (4)			disposal (1)		container (1)						
OOC	can	could	should	would	will	must	might	to	may	'll		
C2V	bins	guzzlers	collector	collection	dispenser	emporium	bucket	eaters	bin	cans		
C2V+embs	can	could	cannot	will	would	should	might	must	*can*	can't		
ELMo	bins	trucks	bags	cans	truck	machines	dump	machine	collection	box		
ELMo+embs	can	cannot	doesn't	could	couldn't	'll	must	don	should	might		
BERT	can	bin	bag	lot	pile	heap	cans	bucket	closet	could		
BERT+embs	can	could	bin	cans	bag	lot	will	may	don	bucket		
ROBERTa	can	cans	bin	will	box	cart	container	could	lot	truck		
ROBERTa+embs	can	could	cans	will	bins	may	container	bin	would	should		
XLNet	can	dump	bin	truck	disposal	pit	heap	pile	container	stand		
XLNet+embs	can	could	will	bin	cannot	dump	may	truck	disposal	stand		
To fight each other is not a natural state for that dog or that bull to be in .												
GOLD	male bovine (1)	male cow (1)	bovine animal (1)	cow (1)								
OOC	bull	bulls	tup	dandie	cow	heifer	goat	nandi	tusker	comyn		
C2V	dog	horse	ought	wants	person	needs	deserves	happens	thing	animal		
C2V+embs	bull	horse	cow	fox	dog	stallion	bulls	cock	lion	pig		
ELMo	person	situation	creature	horse	guy	dog	someone	beast	thing	else		
ELMo+embs	bull	bear	horse	dog	animal	bulls	cat	pet	tiger	beast		
BERT	bull	cow	bitch	beef	ass	devil	cock	ram	wolf	man		
BERT+embs	bull	cow	ram	beef	tiger	cock	buffalo	bitch	horse	elephant		
ROBERTa	bull	bulls	dog	buffalo	animal	man	cow	horse	human	other		
ROBERTa+embs	bull	cat	dog	pig	horse	rabbit	puppy	wolf	bullets	donkey		
XLNet	cat	horse	human	person	man	wolf	lion	pig	bear	cow		
XLNet+embs	bull	horse	cow	lion	pig	cat	bear	tiger	wolf	goat		
Anyway , I saw your ad on the net and just wanted to drop you a line to say hello .												
GOLD	write (3)	email (1)	send a message (1)	send (1)	text (1)	write a letter (1)	send a letter (1)					
OOC	drop	drops	dropping	dropped	drop-off	fall	decrease	decline	slump	pullback		
C2V	give	send	drop	e.mail	ask	leave	post	tell	call	email		
C2V+embs	drop	dropped	dropping	drops	pull	throw	give	send	jump	break		
ELMo	tell	give	send	thank	ask	bring	write	get	throw	follow		
ELMo+embs	drop	jump	send	pull	throw	bring	give	slip	slide	dip		
BERT	drop	give	send	put	cut	dropped	tip	toss	leave	tell		
BERT+embs	drop	drops	dropped	dropping	give	dip	toss	slip	cut	send		
ROBERTa	drop	dropping	dropped	drops	pop	leave	threw	write	send	shoot		
ROBERTa+embs	drop	dropped	dropping	drops	shoot	throw	give	slip	send	toss		
XLNet	drop	send	give	shoot	write	throw	dropped	leave	blow	pop		
XLNet+embs	drop	drops	dropped	dropping	throw	shoot	send	slip	give	fall		

Types of semantic relations: synonym co-hyponym co-hyponym 3 target direct hypernym transitive hypernym

direct hyponym transitive hyponym unknown-relation unknown-word no-path multiword expression

Figure 4: Examples of substitutes produced by various lexical substitution methods based on original neural language models and their improved versions with +embs target word injection. Sentences are from SemEval 2007 dataset.

B Proportions of substitute types

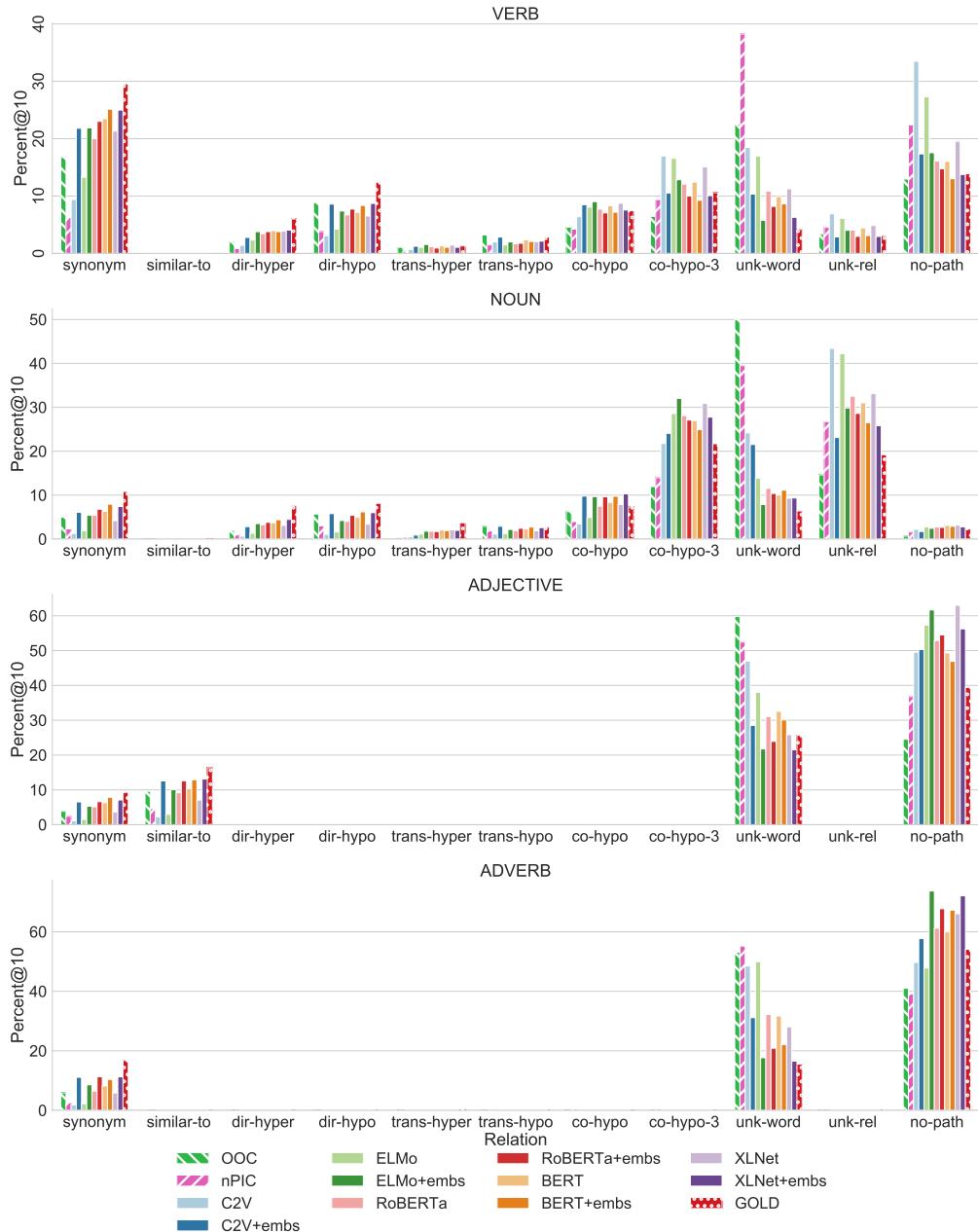


Figure 5: Proportions of substitutes related to the target by various semantic relations according to WordNet. We took top 10 substitutes from each model and all substitutes from the gold standard. Examples are from the CoInCo dataset.

A.3 Paper “Making Fast Graph-based Algorithms with Graph Metric Embeddings”

A. Kutuzov, M. Dorgham, O. Oliynyk, C. Biemann, and A. Panchenko, “**Making Fast Graph-based Algorithms with Graph Metric Embeddings**,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 3349–3355, Association for Computational Linguistics, July 2019

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/P19-1325>

Making Fast Graph-based Algorithms with Graph Metric Embeddings

Andrey Kutuzov[†], Mohammad Dorgham[‡], Oleksiy Oliynyk[‡],
Chris Biemann[‡], and Alexander Panchenko^{*,‡}

[†]Language Technology Group, University of Oslo, Oslo, Norway

[‡]Language Technology Group, Universität Hamburg, Hamburg, Germany

^{*}Skolkovo Institute of Science and Technology, Moscow, Russia

Abstract

The computation of distance measures between nodes in graphs is inefficient and does not scale to large graphs. We explore dense vector representations as an effective way to approximate the same information: we introduce a simple yet efficient and effective approach for learning graph embeddings. Instead of directly operating on the graph structure, our method takes structural measures of pairwise node similarities into account and learns dense node representations reflecting user-defined graph distance measures, such as e.g. the shortest path distance or distance measures that take information beyond the graph structure into account. We demonstrate a speed-up of several orders of magnitude when predicting word similarity by vector operations on our embeddings as opposed to directly computing the respective path-based measures, while outperforming various other graph embeddings on semantic similarity and word sense disambiguation tasks and show evaluations on the WordNet graph and two knowledge base graphs.

When operating on large graphs, such as transportation networks, social networks, or lexical resources, the need for estimating similarities between nodes arises. For many domain-specific applications, custom graph node similarity measures $sim : V \times V \rightarrow \mathbb{R}$ have been defined on pairs of nodes V of a graph $G = (V, E)$. Examples include travel time, communities, or semantic distances for knowledge-based word sense disambiguation on WordNet (Miller, 1995). For instance, the similarity s_{ij} between the cup.n.01 and mug.n.01 synsets in the WordNet is $\frac{1}{4}$ according to the inverted shortest path distance as these two nodes are connected by the undirected path cup → container ← vessel ← drinking_vessel ← mug.

In recent years, a large variety of such node similarity measures have been described, many of

which are based on the notion of a random walk (Fouss et al., 2007; Pilehvar and Navigli, 2015; Lebichot et al., 2018). As given by the structure of the problem, most such measures are defined as traversals of edges E of the graph, which makes their computation prohibitively inefficient.

To this end, we propose the *path2vec* model¹, which solves this problem by decoupling development and use of graph-based measures, and – in contrast to purely walk-based embeddings – is trainable to reflect custom node similarity measures. We represent nodes in a graph with dense embeddings that are good in approximating such custom, e.g. application-specific, pairwise node similarity measures. Similarity computations in a vector space are several orders of magnitude faster than computations directly operating on the graph.

First, effectiveness of our model is shown *intrinsically* by learning metric embeddings for three types of graphs (WordNet, FreeBase, and DBpedia), based on several similarity measures. Second, in an *extrinsic* evaluation on the Word Sense Disambiguation (WSD) task (Navigli, 2009) we replace several original measures with their vectorized counterparts in a known graph-based WSD algorithm by Sinha and Mihalcea (2007), reaching comparable levels of performance with the graph-based algorithms while maintaining computational gains.

The main contribution of this paper is the demonstration of the effectiveness and efficiency of the *path2vec* node embedding method (Kutuzov et al., 2019). This method learns dense vector embeddings of nodes V based on a user-defined custom similarity measure sim , e.g. the shortest path distance or any other similarity measure. While our method is able to closely approximate quite different similarity measures as we show

¹<https://github.com/uhh-lt/path2vec>

on WordNet-based measures and therefore can be used in lieu of these measures in NLP components and applications, our main point is the increase of speed in the similarity computation of nodes, which gains up to 4 orders of magnitude with respect to the original graph-based algorithms.

1 Graph Metric Embeddings Model

Definition of the Model *Path2vec* learns embeddings of the graph nodes $\{v_i, v_j\} \in V$ such that the dot products between pairs of the respective vectors ($\mathbf{v}_i \cdot \mathbf{v}_j$) are close to the user-defined similarities between the nodes s_{ij} . In addition, the model reinforces the similarities $\mathbf{v}_i \cdot \mathbf{v}_n$ and $\mathbf{v}_j \cdot \mathbf{v}_m$ between the nodes v_i and v_j and all their respective adjacent nodes $\{v_n : \exists(v_i, v_n) \in E\}$ and $\{v_m : \exists(v_j, v_m) \in E\}$ to preserve local structure of the graph. The model preserves both **global** and **local** relations between nodes by minimizing $\sum_{(v_i, v_j) \in B} ((\mathbf{v}_i^\top \mathbf{v}_j - s_{ij})^2 - \alpha(\mathbf{v}_i^\top \mathbf{v}_n + \mathbf{v}_j^\top \mathbf{v}_m))$, where $s_{ij} = sim(v_i, v_j)$ is the value of a ‘gold’ similarity measure between a pair of nodes v_i and v_j , \mathbf{v}_i and \mathbf{v}_j are the embeddings of the first and the second node, B is a training batch, α is a regularization coefficient. The second term $(\mathbf{v}_i \cdot \mathbf{v}_n + \mathbf{v}_j \cdot \mathbf{v}_m)$ in the objective function is a regularizer that aids the model to simultaneously maximize the similarity between adjacent nodes while learning the similarity between the two target nodes (one adjacent node is randomly sampled for each target node).

We use negative sampling to form a training batch B adding p negative samples ($s_{ij} = 0$) for each real ($s_{ij} > 0$) training instance: each real node (synset) pair (v_i, v_j) with ‘gold’ similarity s_{ij} is accompanied with p ‘negative’ node pairs (v_i, v_k) and (v_j, v_l) with zero similarities, where v_k and v_l are randomly sampled nodes from V . Embeddings are initialized randomly and trained using the *Adam* optimizer (Kingma and Ba, 2015) with early stopping. Once the model is trained, the computation of node similarities is approximated with the dot product of the learned node vectors, making the computations efficient: $\hat{s}_{ij} = \mathbf{v}_i \cdot \mathbf{v}_j$.

Relation to Similar Models Our model bears resemblance to the Skip-gram model (Mikolov et al., 2013), where the vector dot product $\mathbf{v}_i \cdot \tilde{\mathbf{v}}_j$ of vectors of pairs of words (v_i, v_j) from a training corpus is optimized to a high score close to 1 for observed samples, while the dot products of negative samples are optimized to-

wards 0. In the Skip-gram model, the target is to minimize the log likelihood of the conditional probabilities of context words w_j given current words w_i : $\mathcal{L} = -\sum_{(v_i, v_j) \in B_p} \log \sigma(\mathbf{v}_i \cdot \tilde{\mathbf{v}}_j) - \sum_{(v_i, v_j) \in B_n} \log \sigma(-\mathbf{v}_i \cdot \tilde{\mathbf{v}}_j)$, where B_p is the batch of positive training samples, B_n is the batch of the generated negative samples, and σ is the sigmoid function. At this, Skip-gram uses only **local** information, never creating the full co-occurrence count matrix. In our *path2vec* model, the target dot product values s_{ij} are not binary, but can take arbitrary values in the [0...1] range, as given by the custom distance metric. Further, we use only a single embedding matrix with vector representations of the graph nodes, not needing to distinguish target and context.

Another related model is Global Vectors (GloVe) (Pennington et al., 2014), which learns co-occurrence probabilities in a given corpus. The objective function to be minimized in GloVe model is $\mathcal{L} = \sum_{(v_i, v_j) \in B} f(s_{ij})(\mathbf{v}_i \cdot \tilde{\mathbf{v}}_j - \log s_{ij} + b_i + b_j)^2$, where s_{ij} counts the co-occurrences of words v_i and v_j , b_i and b_j are additional biases for each word, and $f(s_{ij})$ is a weighting function handling rare co-occurrences. Like the Skip-gram, GloVe also uses two embedding matrices, but it relies only on **global** information, pre-aggregating global word co-occurrence counts.

Computing Training Similarities In general case, our model requires computing pairwise node similarities s_{ij} for training between all pairs of nodes in the input graph G . This step could be computationally expensive, but it is done only once to make computing of similarities fast. Besides, for some metrics, effective algorithms exist that compute all pairwise similarities at once, e.g. Johnson (1977) algorithm for computing shortest paths distances with the worst-case performance of $O(|V|^2 \log |V| + |V||E|)$. As the input training dataset also grows quadratically in $|V|$, training time for large graphs can be slow. To address this issue, we found it useful to prune the input training set so that each node $v_i \in V$ has only $k \in [50; 200]$ most similar nodes. Such pruning does not lead to loss of effectiveness.

2 Computational Efficiency

Experimental Setting In this section, we compare efficiency of our method as compared to the original graph based similarity metrics. We

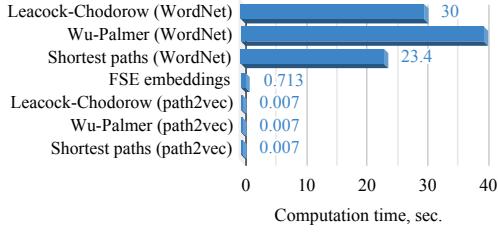


Figure 1: Similarity computation: graph vs vectors.

trained the model on a graph of 82,115 noun synsets from WordNet. Using NLTK (Bird et al., 2009) we computed the following metrics: (1) Leacock-Chodorow similarities (*LCH*) based on the shortest path between two synsets in the WordNet hypernym/hyponym taxonomy and its maximum depth; (2) inverted shortest path distance (*ShP*); (3) Wu-Palmer similarities (*WuP*) based on the depth of the two nodes in the taxonomy and the depth of their most specific ancestor node. For instance, for *LCH* this procedure took about 30 hours on an Intel Xeon E5-2603v4@1.70GHz CPU using 10 threads. We pruned similarities to the first 50 most similar ‘neighbors’ of each synset and trained *path2vec* on this dataset.

Discussion of Results Figure 1 presents computation times for pairwise similarities between one synset and all other 82,115 WordNet noun synsets. We compare running times of calculating two original graph-based metrics to Hamming distance between 128D FSE binary embeddings (Subercaze et al., 2015) and to dot product between their dense vectorized 300D counterparts (using CPU). Using float vectors (*path2vec*) is 4 orders of magnitude faster than operating directly on graphs, and 2 orders faster than Hamming distance. The dot product computation is much faster as compared to shortest path computation (and other complex walks) on a large graph. Also, low-dimensional vector representations of nodes take much less space than the pairwise similarities between all the nodes. The time complexity of calculating the shortest path between graph nodes (as in *ShP* or *LCH*) is in the best case linear in the number of nodes and edges. Calculating Hamming distance between binary strings is linear in the sum of string lengths, which are equivalent of vector sizes (Hamming, 1950). At the same time, the complexity of calculating dot product between float vectors is linear in the vector size and is easily parallelized.

	LCH	ShP	WuP	LCH	ShP	WuP
WordNet	100	100	100	51.3	51.3	47.4
path2vec	93.5	95.2	93.1	53.2	55.5	55.5
TransR	77.6	77.6	72.5		38.6	
node2vec	75.9	75.9	78.7		46.2	
DeepWalk	86.8	86.8	85.0		53.3	
FSE	90.0	90.0	89.0		55.6	

Table 1: Spearman correlations with WordNet similarities (left) and human judgments (right) $\times 100$.

3 Evaluation on Semantic Similarity

Experimental Setting We use noun pairs from the SimLex999 dataset (Hill et al., 2015), measuring Spearman rank correlation between ‘gold’ WordNet distances for these pairs and the vector distances produced by the graph embedding models (trained on WordNet) to see how well the models fit the training objective. We also test the plausibility of the model’s output to human judgments. For this, we use human-annotated similarities from the same SimLex999. Some SimLex999 lemmas can be mapped to more than one WordNet synset. We chose the synset pair with the highest dot product between the embeddings from the corresponding model.

Baselines Our model is compared against five baselines: *raw WordNet similarities* by respective measures; *DeepWalk* (Perozzi et al., 2014); *node2vec* (Grover and Leskovec, 2016); *FSE* (Subercaze et al., 2015); *TransR* (Lin et al., 2015). *DeepWalk*, *node2vec*, and *TransR* models were trained on the same WordNet graph. We used all 82,115 noun synsets as vertices and hypernym/hyponym relations between them as edges. During the training of *DeepWalk* and *node2vec* models, we tested different values for the number of random walks (in the range from 10 to 100), and the vector size (100 to 600). For *DeepWalk*, we additionally experimented with the window size (5 to 100). All other hyperparameters were left at default values. *FSE* embeddings of the WordNet noun synsets were provided to us by the authors, and consist of 128-bit vectors.

Discussion of Results The left part of Table 1 shows results with the WordNet similarity scores used as gold standard. *Path2vec* outperforms other graph embeddings, achieving high correlations with WordNet similarities. This shows that our model efficiently approximates different graph measures. The right part of Table 1 shows results

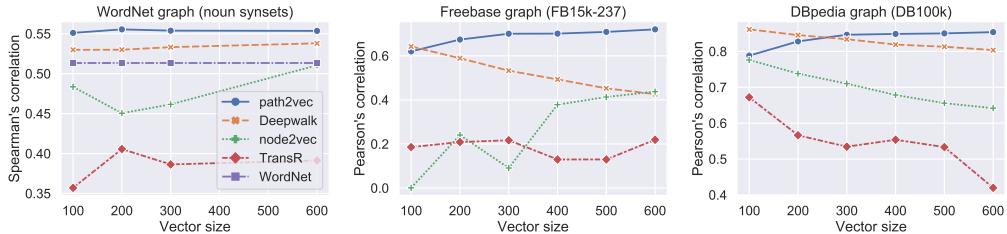


Figure 2: Evaluation on different graphs on SimLex999 (left) and shortest path distance (middle, right).

for the correlations with human judgments (SimLex999). We report the results for the best models for each method, all of them (except *FSE*) using vector size 300 for comparability.

Figure 2 (left) compares *path2vec* to the baselines, as measured by the correlations with SimLex999 human judgments. The WordNet line denotes the correlation of WordNet similarities with SimLex999 scores. For the *path2vec* models, there is a tendency to improve the performance when the vector size is increased (horizontal axis), until a plateau is reached beyond 600. Note that *node2vec* fluctuates, yielding low scores for 200 dimensions. The reported best *DeepWalk* models were trained with the 10 walks and window size 70. The reported best *node2vec* models were trained with 25 walks. Interestingly, *path2vec* and *DeepWalk* models consistently *outperform* the raw WordNet.

4 Evaluation inside a WSD Algorithm

Experimental Setting To showcase how our approach can be used inside a graph-based algorithm, we employ word sense disambiguation (WSD) task, reproducing the approach of (Sinha and Mihalcea, 2007). We replace graph similarities with the dot product between node embeddings and study how it influences the WSD performance. The WSD algorithm starts with building a graph where the nodes are the WordNet synsets of the words in the input sentence. The nodes are then connected by edges weighted with the similarity values between the synset pairs. The final step is selecting the most likely sense for each word based on the weighted in-degree centrality score for each synset.

Discussion of Results Table 2 presents the WSD micro-F1 scores using raw WordNet similarities, 300D *path2vec*, *DeepWalk* and *node2vec* models, and the 128D *FSE* model. We evaluate on the following all-words English WSD test sets:

Model	Senseval2	Senseval3	SemEval-15
Random sense	0.381	0.312	0.393
<i>Graph-based vs vector-based measures</i>			
LCH (WordNet)	0.547	0.494	0.550
LCH (path2vec)	0.527 _{↓0.020}	0.472 _{↓0.022}	0.536 _{↓0.014}
ShP (WordNet)	0.548	0.495	0.550
ShP (path2vec)	0.534 _{↓0.014}	0.489 _{↓0.006}	0.563 _{↑0.013}
WuP (WordNet)	0.547	0.487	0.542
WuP (path2vec)	0.543 _{↓0.004}	0.489 _{↑0.002}	0.545 _{↑0.003}
<i>Various baseline graph embeddings trained on WordNet</i>			
TransR	0.540	0.466	0.536
node2vec	0.503	0.467	0.489
DeepWalk	0.528	0.476	0.552
FSE	0.536	0.476	0.523

Table 2: F1 scores of a graph-based WSD algorithm on WordNet versus its vectorized counterparts.

Senseval-2 (Palmer et al., 2001), Senseval-3 (Mihalcea et al., 2004), and SemEval-15 Task 13 (Moro and Navigli, 2015). The raw WordNet similarities have a small edge over their vector approximations in the majority of the cases yet the *path2vec* models consistently closely follow them while outperforming other graph embedding baselines: We indicate the differences with respect to the original with a subscript number.

5 Evaluation on Knowledge Base Graphs

5.1 Experimental Settings

To show the utility of our model besides the WordNet graph, we also applied it to two graphs derived from knowledge bases (KBs). More specifically, we base our experiments on two publicly available standard samples from these two resources: the FB15k-237 (Toutanova and Chen, 2015) dataset contains 14,951 entities/nodes and is derived from Freebase (Bollacker et al., 2008); the DB100k (Ding et al., 2018) dataset contains 99,604 entities/nodes and is derived from DBPe-

dia (Auer et al., 2007).

It is important to note that both datasets were used to evaluate approaches that learn knowledge graph embeddings, e.g. (Lin et al., 2015; Xie et al., 2016; Joulin et al., 2017) on the task on *knowledge base completion* (KBC), to predict missing KB edges/relations between nodes/entities. The specificity of our model is that it learns a given graph similarity metric, which is not provided in these datasets. Therefore, we use only the graphs from these datasets, computing the shortest path distances between all pairs of nodes using the algorithm of Johnson (1977). Instead of the KBC task, we evaluate on the task of predicting node similarity, here using the shortest path distance. We generate a random sample of node pairs for testing from the set of all node pairs (these pairs are excluded from training). The test set contains an equal number of paths of length 1-7 (in total 1050 pairs each, 150 pairs per path length).

5.2 Discussion of Results

Figure 2 (middle and right) shows evaluation results on the knowledge base graphs. *Path2vec* is able to better approximate the target graph metric than the standard graph embedding models. As dimensionality of the embeddings increases, the model more closely approximates the target metric, but the performance drop for the models with a low number of dimensions is not drastic, allowing more effective computations while maintaining a reasonable efficiency level. Regarding the competitors, DeepWalk comes closest to the performance of our approach, but does not seem to make use of the additional dimensions when training on larger vector sizes; on the DBpedia dataset, this issue is shared between all baselines, where correlation to the true path lengths decreases as representation length increases.

6 Related Work

Representation learning on graphs received much attention recently in various research communities, see Hamilton et al. (2017a) for a thorough survey on the existing methods. All of them (including ours) are based on the idea of projecting graph nodes into a latent space with a much lower dimensionality than the number of nodes. Existing approaches to graph embeddings use either factorization of the graph adjacency matrix (Cao et al., 2015; Ou et al., 2016) or random

walks over the graph as in *Deepwalk* (Perozzi et al., 2014) and *node2vec* (Grover and Leskovec, 2016). A different approach is taken by Subercaze et al. (2015), who directly embed the WordNet tree graph into Hamming hypercube binary representations. Their ‘Fast similarity embedding’ (FSE) model provides a quick way of calculating semantic similarities based on WordNet. The FSE embeddings are not differentiable though, considerably limiting their use in deep neural architectures. *TransR* (Lin et al., 2015) extends *TransH* (Wang et al., 2014) and is based on the idea that an entity may have a few aspects and different relations are focused on them. So the same entities can be close or far from each other depending on the type of the relation. *TransR* projects entity vectors into a relation specific space, and learns embeddings via translation between projected entities.

We compare our *path2vec* model to these approaches, yet we did not compare to the models like *GraphSAGE* embeddings (Hamilton et al., 2017b) and Graph Convolutional Networks (Schlichtkrull et al., 2018) as they use node features which are absent in our setup.

7 Conclusion

Structured knowledge contained in language networks is useful for NLP applications but is difficult to use directly in neural architectures. We proposed a way to train embeddings that directly represent a graph-based similarity measure structure. Our model, *path2vec*, relies on both global and local information from the graph and is simple, effective, and computationally efficient. We demonstrated that our approach generalizes well across graphs (WordNet, Freebase, and DBpedia). Besides, we integrated it into a graph-based WSD algorithm, showing that its vectorized counterpart yields comparable F1 scores on three datasets.

Path2vec enables a speed-up of up to four orders of magnitude for the computation of graph distances as compared to ‘direct’ graph measures. Thus, our model is simple and general, hence it may be applied to any graph together with a node distance measure to speed up algorithms that employ graph distances.

Acknowledgments

This was supported by the DFG under “JOIN-T” (BI 1544/4) and “ACQuA” (BI 1544/7) projects.

References

- Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. *DBpedia: A nucleus for a web of open data*. In *The Semantic Web: Proceedings of the 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007*, pages 722–735, Busan, South Korea. Springer.
- Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.
- Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. *Freebase: a collaboratively created graph database for structuring human knowledge*. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250, Vancouver, BC, Canada. ACM.
- Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. *GraRep: Learning graph representations with global structural information*. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management*, pages 891–900, Melbourne, Australia. ACM.
- Boyang Ding, Quan Wang, Bin Wang, and Li Guo. 2018. *Improving knowledge graph embedding using simple constraints*. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 110–121, Melbourne, Australia. Association for Computational Linguistics.
- Francois Fouss, Alain Pirotte, Jean-Michel Renders, and Marco Saerens. 2007. *Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation*. *IEEE Transactions on knowledge and data engineering*, 19(3):355–369.
- Aditya Grover and Jure Leskovec. 2016. *Node2vec: Scalable feature learning for networks*. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, San Francisco, CA, USA. ACM.
- William Hamilton, Rex Ying, and Jure Leskovec. 2017a. *Representation learning on graphs: Methods and applications*. *IEEE Data Engineering Bulletin*, 40(3):52–74.
- William Hamilton, Zhitao Ying, and Jure Leskovec. 2017b. *Inductive representation learning on large graphs*. In *Advances in Neural Information Processing Systems*, pages 1024–1034, Long Beach, CA, USA.
- Richard Hamming. 1950. *Error detecting and error correcting codes*. *Bell System technical journal*, 29(2):147–160.
- Felix Hill, Roi Reichart, and Anna Korhonen. 2015. *SimLex-999: Evaluating Semantic Models With (Genuine) Similarity Estimation*. *Computational Linguistics*, 41(4):665–695.
- Donald B. Johnson. 1977. *Efficient algorithms for shortest paths in sparse networks*. *Journal of the ACM (JACM)*, 24(1):1–13.
- Armand Joulin, Edouard Grave, Piotr Bojanowski, Maximilian Nickel, and Tomas Mikolov. 2017. *Fast linear model for knowledge graph embeddings*. *arXiv preprint arXiv:1710.10881*.
- Diederik P. Kingma and Jimmy Ba. 2015. *Adam: A method for stochastic optimization*. In *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, USA.
- Andrey Kutuzov, Mohammad Dorgham, Oleksiy Oliynyk, Chris Biemann, and Alexander Panchenko. 2019. *Learning graph embeddings from WordNet-based similarity measures*. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 125–135, Minneapolis, MN, USA. Association for Computational Linguistics.
- Bertrand Lebichot, Guillaume Guex, Ilkka Kivimäki, and Marco Saerens. 2018. *A constrained randomized shortest-paths framework for optimal exploration*. *arXiv preprint arXiv:1807.04551*.
- Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. *Learning entity and relation embeddings for knowledge graph completion*. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 2181–2187, Austin, TX, USA. AAAI Press.
- Rada Mihalcea, Timothy Chklovski, and Adam Kilgarriff. 2004. *The Senseval-3 English lexical sample task*. In *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 25–28, Barcelona, Spain. Association for Computational Linguistics.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. *Distributed representations of words and phrases and their compositionality*. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119, Lake Tahoe, NV, USA. Curran Associates, Inc.
- George A. Miller. 1995. *WordNet: A lexical database for English*. *Communications of the ACM*, 38(11):39–41.
- Andrea Moro and Roberto Navigli. 2015. *Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking*. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 288–297. Association for Computational Linguistics.

- Roberto Navigli. 2009. [Word sense disambiguation: A survey](#). *ACM Computing Surveys (CSUR)*, 41(2):10.
- Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. [Asymmetric transitivity preserving graph embedding](#). In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1105–1114, San Francisco, CA, USA. ACM.
- Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. 2001. [English tasks: All-words and verb lexical sample](#). In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 21–24, Toulouse, France. Association for Computational Linguistics.
- Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. [Glove: Global vectors for word representation](#). In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.
- Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. [DeepWalk: Online learning of social representations](#). In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, New York, NY, USA. ACM.
- Mohammad T. Pilehvar and Roberto Navigli. 2015. [From senses to texts: An all-in-one graph-based approach for measuring semantic similarity](#). *Artificial Intelligence*, 228:95–128.
- Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. [Modeling relational data with graph convolutional networks](#). In *Proceedings of the European Semantic Web Conference 2018: The Semantic Web*, pages 593–607, Heraklion, Greece. Springer.
- Ravi Sinha and Rada Mihalcea. 2007. [Unsupervised graph-based word sense disambiguation using measures of word semantic similarity](#). In *International Conference on Semantic Computing (ICSC)*, pages 363–369, Irvine, CA, USA. IEEE.
- Julien Subercaze, Christophe Gravier, and Frédérique Laforest. 2015. [On metric embedding for boosting semantic similarity computations](#). In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 8–14, Beijing, China. Association for Computational Linguistics.
- Kristina Toutanova and Danqi Chen. 2015. [Observed versus latent features for knowledge base and text inference](#). In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, Beijing, China. Association for Computational Linguistics.
- Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. [Knowledge graph embedding by translating on hyperplanes](#). In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 1112–1119, Québec City, QC, Canada. AAAI Press.
- Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. [Representation learning of knowledge graphs with entity descriptions](#). In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 2659–2665, Phoenix, AZ, USA. AAAI Press.

A.4 Paper “Negative Sampling Improves Hypernymy Extraction Based on Projection Learning”

D. Ustalov, N. Arefyev, C. Biemann, and A. Panchenko, “**Negative Sampling Improves Hypernymy Extraction Based on Projection Learning**,” in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, (Valencia, Spain), pp. 543–550, Association for Computational Linguistics, Apr. 2017

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/E17-2087>

Negative Sampling Improves Hypernymy Extraction Based on Projection Learning

Dmitry Ustalov[†], Nikolay Arefyev[§], Chris Biemann[‡], and Alexander Panchenko[‡]

[†]Ural Federal University, Institute of Natural Sciences and Mathematics, Russia

[§]Moscow State University, Faculty of Computational Mathematics and Cybernetics, Russia

[‡]University of Hamburg, Department of Informatics, Language Technology Group, Germany

dmitry.ustalov@urfu.ru, narefjev@cs.msu.ru

{biemann, panchenko}@informatik.uni-hamburg.de

Abstract

We present a new approach to extraction of hypernyms based on projection learning and word embeddings. In contrast to classification-based approaches, projection-based methods require no candidate hyponym-hypernym pairs. While it is natural to use both positive and negative training examples in supervised relation extraction, the impact of negative examples on hypernym prediction was not studied so far. In this paper, we show that explicit negative examples used for regularization of the model significantly improve performance compared to the state-of-the-art approach of Fu et al. (2014) on three datasets from different languages.

1 Introduction

Hypernyms are useful in many natural language processing tasks ranging from construction of taxonomies (Snow et al., 2006; Panchenko et al., 2016a) to query expansion (Gong et al., 2005) and question answering (Zhou et al., 2013). Automatic extraction of hypernyms from text has been an active area of research since manually constructed high-quality resources featuring hypernyms, such as WordNet (Miller, 1995), are not available for many domain-language pairs.

The drawback of pattern-based approaches to hypernymy extraction (Hearst, 1992) is their sparsity. Approaches that rely on the classification of pairs of word embeddings (Levy et al., 2015) aim to tackle this shortcoming, but they require candidate hyponym-hypernym pairs. We explore a hypernymy extraction approach that requires no candidate pairs. Instead, the method performs prediction of a hypernym embedding on the basis of a hyponym embedding.

The contribution of this paper is a novel approach for hypernymy extraction based on projection learning. Namely, we present an improved version of the model proposed by Fu et al. (2014), which makes use of both positive and negative training instances enforcing the asymmetry of the projection. The proposed model is generic and could be straightforwardly used in other relation extraction tasks where both positive and negative training samples are available. Finally, we are the first to successfully apply projection learning for hypernymy extraction in a morphologically rich language. An implementation of our approach and the pre-trained models are available online.¹

2 Related Work

Path-based methods for hypernymy extraction rely on sentences where both hyponym and hypernym co-occur in characteristic contexts, e.g., “such cars as *Mercedes* and *Audi*”. Hearst (1992) proposed to use hand-crafted lexical-syntactic patterns to extract hypernyms from such contexts. Snow et al. (2004) introduced a method for learning patterns automatically based on a set of seed hyponym-hypernym pairs. Further examples of path-based approaches include (Tjong Kim Sang and Hofmann, 2009) and (Navigli and Velardi, 2010). The inherent limitation of the path-based methods leading to sparsity issues is that hyponym and hypernym have to co-occur in the same sentence.

Methods based on distributional vectors, such as those generated using the *word2vec* toolkit (Mikolov et al., 2013b), aim to overcome this sparsity issue as they require no hyponym-hypernym co-occurrence in a sentence. Such methods take representations of individual words as an input to predict relations between them.

¹<http://github.com/nlpub/projlearn>

Two branches of methods relying on distributional representations emerged so far.

Methods based on word pair classification take an ordered pair of word embeddings (a candidate hyponym-hypernym pair) as an input and output a binary label indicating a presence of the hypernymy relation between the words. Typically, a binary classifier is trained on concatenation or subtraction of the input embeddings, cf. (Roller et al., 2014). Further examples of such methods include (Lenci and Benotto, 2012; Weeds et al., 2014; Levy et al., 2015; Vylomova et al., 2016).

HypeNET (Shwartz et al., 2016) is a hybrid approach which is also based on a classifier, but in addition to two word embeddings a third vector is used. It represents path-based syntactic information encoded using an LSTM model (Hochreiter and Schmidhuber, 1997). Their results significantly outperform the ones from previous path-based work of Snow et al. (2004).

An inherent limitation of classification-based approaches is that they require a list of candidate words pairs. While these are given in evaluation datasets such as BLESS (Baroni and Lenci, 2011), a corpus-wide classification of relations would need to classify all possible word pairs, which is computationally expensive for large vocabularies. Besides, Levy et al. (2015) discovered a tendency to lexical memorization of such approaches hampering the generalization.

Methods based on projection learning take one hyponym word vector as an input and output a word vector in a topological vicinity of hypernym word vectors. Scaling this to the vocabulary, there is only one such operation per word. Mikolov et al. (2013a) used projection learning for bilingual word translation. Vulić and Korhonen (2016) presented a systematic study of four classes of methods for learning bilingual embeddings including those based on projection learning.

Fu et al. (2014) were first to apply projection learning for hypernym extraction. Their approach is to learn an affine transformation of a hyponym into a hypernym word vector. The training of their model is performed with stochastic gradient descent. The k -means clustering algorithm is used to split the training relations into several groups. One transformation is learned for each group, which can account for the possibility that the projection of the relation depends on a subspace. This state-of-the-art approach serves as the baseline in our

experiments.

Nayak (2015) performed evaluations of distributional hypernym extractors based on classification and projection methods (yet on different datasets, so these approaches are not directly comparable). The best performing projection-based architecture proposed in this experiment is a four-layered feed-forward neural network. No clustering of relations was used. The author used negative samples in the model by adding a regularization term in the loss function. However, drawing negative examples uniformly from the vocabulary turned out to hamper performance. In contrast, our approach shows significant improvements using manually created synonyms and hyponyms as negative samples.

Yamane et al. (2016) introduced several improvements of the model of Fu et al. (2014). Their model jointly learns projections and clusters by dynamically adding new clusters during training. They also used automatically generated negative instances via a regularization term in the loss function. In contrast to Nayak (2015), negative samples are selected not randomly, but among nearest neighbors of the predicted hypernym. Their approach compares favorably to (Fu et al., 2014), yet the contribution of the negative samples was not studied. Key differences of our approach from (Yamane et al., 2016) are (1) use of explicit as opposed to automatically generated negative samples, (2) enforcement of asymmetry of the projection matrix via re-projection. While our experiments are based on the model of Fu et al. (2014), our regularizers can be straightforwardly integrated into the model of Yamane et al. (2016).

3 Hypernymy Extraction via Regularized Projection Learning

3.1 Baseline Approach

In our experiments, we use the model of Fu et al. (2014) as the baseline. In this approach, the projection matrix Φ^* is obtained similarly to the linear regression problem, i.e., for the given row word vectors x and y representing correspondingly hyponym and hypernym, the square matrix Φ^* is fit on the training set of positive pairs \mathcal{P} :

$$\Phi^* = \arg \min_{\Phi} \frac{1}{|\mathcal{P}|} \sum_{(x,y) \in \mathcal{P}} \|x\Phi - y\|^2,$$

where $|\mathcal{P}|$ is the number of training examples and $\|x\Phi - y\|$ is the distance between a pair of row

vectors $\mathbf{x}\Phi$ and \mathbf{y} . In the original method, the L^2 distance is used. To improve performance, k projection matrices Φ are learned one for each cluster of relations in the training set. One example is represented by a hyponym-hypernym offset. Clustering is performed using the k -means algorithm (MacQueen, 1967).

3.2 Linguistic Constraints via Regularization

The nearest neighbors generated using distributional word vectors tend to contain a mixture of synonyms, hypernyms, co-hyponyms and other related words (Wandmacher, 2005; Heylen et al., 2008; Panchenko, 2011). In order to explicitly provide examples of undesired relations to the model, we propose two improved versions of the baseline model: *asymmetric regularization* that uses inverted relations as negative examples, and *neighbor regularization* that uses relations of other types as negative examples. For that, we add a regularization term to the loss function:

$$\Phi^* = \arg \min_{\Phi} \frac{1}{|\mathcal{P}|} \sum_{(\mathbf{x}, \mathbf{y}) \in \mathcal{P}} \|\mathbf{x}\Phi - \mathbf{y}\|^2 + \lambda R,$$

where λ is the constant controlling the importance of the regularization term R .

Asymmetric Regularization. As hypernymy is an asymmetric relation, our first method enforces the asymmetry of the projection matrix. Applying the same transformation to the predicted hypernym vector $\mathbf{x}\Phi$ should not provide a vector similar (\cdot) to the initial hyponym vector \mathbf{x} . Note that, this regularizer requires only positive examples \mathcal{P} :

$$R = \frac{1}{|\mathcal{P}|} \sum_{(\mathbf{x}, \cdot) \in \mathcal{P}} (\mathbf{x}\Phi\Phi \cdot \mathbf{x})^2.$$

Neighbor Regularization. This approach relies on the negative sampling by explicitly providing the examples of semantically related words \mathbf{z} of the hyponym \mathbf{x} that penalizes the matrix to produce the vectors similar to them:

$$R = \frac{1}{|\mathcal{N}|} \sum_{(\mathbf{x}, \mathbf{z}) \in \mathcal{N}} (\mathbf{x}\Phi\Phi \cdot \mathbf{z})^2.$$

Note that this regularizer requires negative samples \mathcal{N} . In our experiments, we use synonyms of hyponyms as \mathcal{N} , but other types of relations can be also used such as antonyms, meronyms or co-hyponyms. Certain words might have no synonyms in the training set. In such cases, we substitute \mathbf{z} with \mathbf{x} , gracefully reducing to the previous variation. Otherwise, on each training epoch, we sample a random synonym of the given word.

Regularizers without Re-Projection. In addition to the two regularizers described above, that rely on re-projection of the hyponym vector ($\mathbf{x}\Phi\Phi$), we also tested two regularizers without re-projection, denoted as $\mathbf{x}\Phi$. The neighbor regularizer in this variation is defined as follows:

$$R = \frac{1}{|\mathcal{N}|} \sum_{(\mathbf{x}, \mathbf{z}) \in \mathcal{N}} (\mathbf{x}\Phi \cdot \mathbf{z})^2.$$

In our case, this regularizer penalizes relatedness of the predicted hypernym $\mathbf{x}\Phi$ to the synonym \mathbf{z} . The asymmetric regularizer without re-projection is defined in a similar way.

3.3 Training of the Models

To learn parameters of the considered models we used the Adam method (Kingma and Ba, 2014) with the default meta-parameters as implemented in the TensorFlow framework (Abadi et al., 2016).² We ran 700 training epochs passing a batch of 1024 examples to the optimizer. We initialized elements of each projection matrix using the normal distribution $\mathcal{N}(0, 0.1)$.

4 Results

4.1 Evaluation Metrics

In order to assess the quality of the model, we adopted the $\text{hit}@l$ measure proposed by Frome et al. (2013) which was originally used for image tagging. For each subsumption pair (\mathbf{x}, \mathbf{y}) composed of the hyponym \mathbf{x} and the hypernym \mathbf{y} in the test set \mathcal{P} , we compute l nearest neighbors for the projected hypernym $\mathbf{x}\Phi^*$. The pair is considered matched if the gold hypernym \mathbf{y} appears in the computed list of the l nearest neighbors $\text{NN}_l(\mathbf{x}\Phi^*)$. To obtain the quality score, we average the matches in the test set \mathcal{P} :

$$\text{hit}@l = \frac{1}{|\mathcal{P}|} \sum_{(\mathbf{x}, \mathbf{y}) \in \mathcal{P}} \mathbb{1}(\mathbf{y} \in \text{NN}_l(\mathbf{x}\Phi^*)),$$

where $\mathbb{1}(\cdot)$ is the indicator function. To consider also the rank of the correct answer, we compute the area under curve measure as the area under the $l - 1$ trapezoids:

$$\text{AUC} = \frac{1}{2} \sum_{i=1}^{l-1} (\text{hit}@(i) + \text{hit}@(i+1)).$$

4.2 Experiment 1: The Russian Language

Dataset. In this experiment, we use word embeddings published as a part of the Russian Dis-

²<https://www.tensorflow.org>

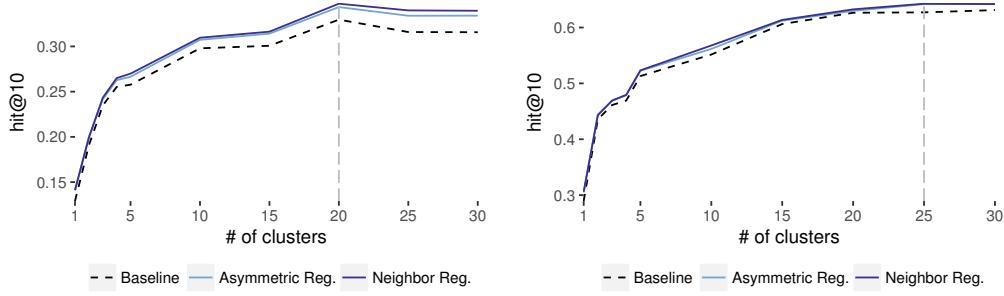


Figure 1: Performance of our models with re-projection as compared to the baseline approach of (Fu et al., 2014) according to the hit@10 measure for Russian (left) and English (right) on the validation set.

Model	hit@1	hit@5	hit@10	AUC
Baseline	0.209	0.303	0.323	2.665
Asym. Reg. $\alpha\Phi$	0.213	0.300	0.322	2.659
Asym. Reg. $\alpha\Phi\Phi$	0.212	0.312	0.334	2.743
Neig. Reg. $\alpha\Phi$	0.214	0.304	0.325	2.685
Neig. Reg. $\alpha\Phi\Phi$	0.211	0.315	0.338	2.768

Table 1: Performance of our approach for Russian for $k = 20$ clusters compared to (Fu et al., 2014).

tributional Thesaurus (Panchenko et al., 2016b) trained on 12.9 billion token collection of Russian books. The embeddings were trained using the skip-gram model (Mikolov et al., 2013b) with 500 dimensions and a context window of 10 words.

The dataset used in our experiments has been composed of two sources. We extracted synonyms and hypernyms from the Wiktionary³ using the Wikokit toolkit (Krizhanovsky and Smirnov, 2013). To enrich the lexical coverage of the dataset, we extracted additional hypernyms from the same corpus using Hearst patterns for Russian using the PatternSim toolkit (Panchenko et al., 2012).⁴ To filter noisy extractions, we used only relations extracted more than 100 times.

As suggested by Levy et al. (2015), we split the train and test sets such that each contains a distinct vocabulary to avoid the lexical overfitting. This results in 25 067 training, 8 192 validation, and 8 310 test examples. The validation and test sets contain hypernyms from Wiktionary, while the training set is composed of hypernyms and synonyms coming from both sources.

Discussion of Results. Figure 1 (left) shows performance of the three projection learning setups on the validation set: the baseline approach, the asymmetric regularization approach, and the

neighbor regularization approach. Both regularization strategies lead to consistent improvements over the non-regularized baseline of (Fu et al., 2014) across various cluster sizes. The method reaches optimal performance for $k = 20$ clusters. Table 1 provides a detailed comparison of the performance metrics for this setting. Our approach based on the regularization using synonyms as negative samples outperform the baseline (all differences between the baseline and our models are significant with respect to the t -test). According to all metrics, but hit@1 for which results are comparable to $\alpha\Phi$, the re-projection ($\alpha\Phi\Phi$) improves results.

4.3 Experiment 2: The English Language

We performed the evaluation on two datasets.

EVALution Dataset. In this evaluation, word embeddings were trained on a 6.3 billion token text collection composed of Wikipedia, ukWaC (Ferraresi et al., 2008), Gigaword (Graff, 2003), and news corpora from the Leipzig Collection (Goldhahn et al., 2012). We used the skip-gram model with the context window size of 8 tokens and 300-dimensional vectors.

We use the EVALution dataset (Santus et al., 2015) for training and testing the model, composed of 1 449 hypernyms and 520 synonyms, where hypernyms are split into 944 training, 65 validation and 440 test pairs. Similarly to the first experiment, we extracted extra training hypernyms using the Hearst patterns, but in contrast to Russian, they did not improve the results significantly, so we left them out for English. A reason for such difference could be the more complex morphological system of Russian, where each word has more morphological variants compared

³<http://www.wiktionary.org>

⁴<https://github.com/cental/patternsim>

Model	k	EVALution				EVALution, BLESS, K&H+N, ROOT09				
		hit@1	hit@5	hit@10	AUC	hit@1	hit@5	hit@10	AUC	
Baseline	1	0.109	0.118	0.120	1.052	1	0.104	0.247	0.290	2.115
Asymmetric Reg. $x\Phi$	1	0.116	0.125	0.132	1.140	1	0.132	0.256	0.292	2.204
Asymmetric Reg. $x\Phi\Phi$	1	0.145	0.166	0.173	1.466	1	0.112	0.266	0.314	2.267
Neighbor Reg. $x\Phi$	1	0.134	0.141	0.150	1.280	1	0.134	0.255	0.306	2.267
Neighbor Reg. $x\Phi\Phi$	1	0.148	0.168	0.177	1.494	1	0.111	0.264	0.316	2.273
Baseline	30	0.327	0.339	0.350	3.080	25	0.546	0.614	0.634	5.481
Asymmetric Reg. $x\Phi$	30	0.336	0.354	0.366	3.201	25	0.547	0.616	0.632	5.492
Asymmetric Reg. $x\Phi\Phi$	30	0.341	0.364	0.368	3.255	25	0.553	0.621	0.642	5.543
Neighbor Reg. $x\Phi$	30	0.339	0.357	0.364	3.210	25	0.547	0.617	0.634	5.494
Neighbor Reg. $x\Phi\Phi$	30	0.345	0.366	0.370	3.276	25	0.553	0.623	0.641	5.547

Table 2: Performance of our approach for English without clustering ($k = 1$) and with the optimal number of cluster on the EVALution datasets ($k = 30$) and on the combined datasets ($k = 25$).

to English. Therefore, extra training samples are needed for Russian (embeddings of Russian were trained on a non-lemmatized corpus).

Combined Dataset. To show the robustness of our approach across configurations, this dataset has more training instances, different embeddings, and both synonyms and co-hyponyms as negative samples. We used hypernyms, synonyms and co-hyponyms from the four commonly used datasets: EVALution, BLESS (Baroni and Lenci, 2011), ROOT09 (Santus et al., 2016) and K&H+N (Neculescu et al., 2015). The obtained 14 528 relations were split into 9 959 training, 1 631 validation and 1 625 test hypernyms; 1 313 synonyms and co-hyponyms were used as negative samples. We used the standard 300-dimensional embeddings trained on the 100 billion tokens Google News corpus (Mikolov et al., 2013b).

Discussion of Results. Figure 1 (right) shows that similarly to Russian, both regularization strategies lead to consistent improvements over the non-regularized baseline. Table 2 presents detailed results for both English datasets. Similarly to the first experiment, our approach consistently improves results robustly across various configurations. As we change the number of clusters, types of embeddings, the size of the training data and type of relations used for negative sampling, results using our method stay superior to those of the baseline. The regularizers without re-projection ($x\Phi$) obtain lower results in most configurations as compared to re-projected versions ($x\Phi\Phi$). Overall, the neighbor regularization yields slightly better results in comparison to the asymmetric regularization. We attribute this to the fact that some synonyms z are close to the original hyponym x , while others can be distant. Thus, neighbor regularization is able to safeguard

the model during training from more errors. This is also a likely reason why the performance of both regularizers is similar: the asymmetric regularization makes sure that a re-projected vector does not belong to a semantic neighborhood of the hyponym. Yet, this is exactly what neighbor regularization achieves. Note, however that neighbor regularization requires explicit negative examples, while asymmetric regularization does not.

5 Conclusion

In this study, we presented a new model for extraction of hypernymy relations based on the projection of distributional word vectors. The model incorporates information about explicit negative training instances represented by relations of other types, such as synonyms and co-hyponyms, and enforces asymmetry of the projection operation. Our experiments in the context of the hypernymy prediction task for English and Russian languages show significant improvements of the proposed approach over the state-of-the-art model without negative sampling.

Acknowledgments

We acknowledge the support of the Deutsche Forschungsgemeinschaft (DFG) foundation under the “JOIN-T” project, the Deutscher Akademischer Austauschdienst (DAAD), the Russian Foundation for Basic Research (RFBR) under the project no. 16-37-00354 mol_a, and the Russian Foundation for Humanities under the project no. 16-04-12019 “RussNet and YARN thesauri integration”. We also thank Microsoft for providing computational resources under the Microsoft Azure for Research award. Finally, we are grateful to Benjamin Milde, Andrey Kutuzov, Andrew Krizhanovsky, and Martin Riedl for discussions and suggestions related to this study.

References

- Martín Abadi et al. 2016. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *CoRR*, abs/1603.04467.
- Marco Baroni and Alessandro Lenci. 2011. How We BLESSED Distributional Semantic Evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, GEMS '11, pages 1–10, Edinburgh, Scotland. Association for Computational Linguistics.
- Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukWaC, a very large Web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4): Can we beat Google?*, pages 47–54, Marakech, Morocco.
- Andrea Frome, Greg S. Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc’ Aurelio Ranzato, and Tomas Mikolov. 2013. DeViSE: A Deep Visual-Semantic Embedding Model. In *Advances in Neural Information Processing Systems 26*, pages 2121–2129. Curran Associates, Inc., Harrahs and Harveys, NV, USA.
- Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Learning Semantic Hierarchies via Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1199–1209, Baltimore, MD, USA. Association for Computational Linguistics.
- Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building Large Monolingual Dictionaries at the Leipzig Corpora Collection: From 100 to 200 Languages. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 759–765, Istanbul, Turkey. European Language Resources Association (ELRA).
- Zhiguo Gong, Chan Wa Cheang, and U. Leong Hou. 2005. Web Query Expansion by WordNet. In *Proceedings of the 16th International Conference on Database and Expert Systems Applications - DEXA '05*, pages 166–175. Springer Berlin Heidelberg, Copenhagen, Denmark.
- David Graff. 2003. English Gigaword. Technical Report LDC2003T05, Linguistic Data Consortium, Philadelphia, PA, USA.
- Marti A. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of the 14th Conference on Computational Linguistics - Volume 2*, COLING'92, pages 539–545, Nantes, France. Association for Computational Linguistics.
- Kris Heylen, Yves Peirsman, Dirk Geeraerts, and Dirk Speelman. 2008. Modelling Word Similarity: an Evaluation of Automatic Synonymy Extraction Algorithms. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, pages 3243–3249, Marrakech, Morocco. European Language Resources Association (ELRA).
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.
- Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980.
- Andrew A. Krizhanovsky and Alexander V. Smirnov. 2013. An approach to automated construction of a general-purpose lexical ontology based on Wiktionary. *Journal of Computer and Systems Sciences International*, 52(2):215–225.
- Alessandro Lenci and Giulia Benotto. 2012. Identifying Hypernyms in Distributional Semantic Spaces. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, pages 75–79, Montréal, Canada. Association for Computational Linguistics.
- Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. 2015. Do Supervised Distributional Methods Really Learn Lexical Inference Relations? In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 970–976, Denver, Colorado, USA. Association for Computational Linguistics.
- James MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, California, USA. University of California Press.
- Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013a. Exploiting Similarities among Languages for Machine Translation. *CoRR*, abs/1309.4168.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013b. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., Harrahs and Harveys, NV, USA.
- George A. Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41.
- Roberto Navigli and Paola Velardi. 2010. Learning Word-Class Lattices for Definition and Hyponym Extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1318–1327, Uppsala, Sweden. Association for Computational Linguistics.

- Neha Nayak. 2015. Learning Hypernymy over Word Embeddings. Technical report, Stanford University.
- Silvia Neculescu, Sara Mendes, David Jurgens, Núria Bel, and Roberto Navigli. 2015. Reading Between the Lines: Overcoming Data Sparsity for Accurate Classification of Lexical Relationships. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 182–192, Denver, CO, USA. Association for Computational Linguistics.
- Alexander Panchenko, Olga Morozova, and Hubert Naets. 2012. A Semantic Similarity Measure Based on Lexico-Syntactic Patterns. In *Proceedings of KONVENS 2012*, pages 174–178, Vienna, Austria. ÖGAI.
- Alexander Panchenko, Stefano Faralli, Eugen Ruppert, Steffen Remus, Hubert Naets, Cedrick Fairon, Simone Paolo Ponazzo, and Chris Biemann. 2016a. TAXI at SemEval-2016 Task 13: a Taxonomy Induction Method based on Lexico-Syntactic Patterns, Substrings and Focused Crawling. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1320–1327, San Diego, CA, USA. Association for Computational Linguistics.
- Alexander Panchenko, Dmitry Ustalov, Nikolay Arefyev, Denis Paperno, Natalia Konstantinova, Natalia Loukachevitch, and Chris Biemann. 2016b. Human and Machine Judgements for Russian Semantic Relatedness. In *Proceedings of the 5th Conference on Analysis of Images, Social Networks and Texts (AIST'2016)*, volume 661 of *Communications in Computer and Information Science*, pages 303–317, Yekaterinburg, Russia. Springer-Verlag Berlin Heidelberg.
- Alexander Panchenko. 2011. Comparison of the Baseline Knowledge-, Corpus-, and Web-based Similarity Measures for Semantic Relations Extraction. In *Proceedings of the GEMS 2011 Workshop on Geometrical Models of Natural Language Semantics*, pages 11–21, Edinburgh, UK. Association for Computational Linguistics.
- Stephen Roller, Katrin Erk, and Gemma Boleda. 2014. Inclusive yet Selective: Supervised Distributional Hypernymy Detection. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1025–1036, Dublin, Ireland, August. Dublin City University and Association for Computational Linguistics.
- Enrico Santus, Frances Yung, Alessandro Lenci, and Chu-Ren Huang. 2015. EVALution 1.0: an Evolving Semantic Dataset for Training and Evaluation of Distributional Semantic Models. In *Proceedings of the 4th Workshop on Linked Data in Linguistics: Resources and Applications*, pages 64–69, Beijing, China. Association for Computational Linguistics.
- Enrico Santus, Alessandro Lenci, Tin-Shing Chiu, Qin Lu, and Chu-Ren Huang. 2016. Nine Features in a Random Forest to Learn Taxonomical Semantic Relations. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 4557–4564, Portorož, Slovenia. European Language Resources Association (ELRA).
- Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving Hypernymy Detection with an Integrated Path-based and Distributional Method. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2389–2398, Berlin, Germany. Association for Computational Linguistics.
- Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2004. Learning Syntactic Patterns for Automatic Hypernym Discovery. In *Proceedings of the 17th International Conference on Neural Information Processing Systems, NIPS'04*, pages 1297–1304, Vancouver, British Columbia, Canada. MIT Press.
- Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2006. Semantic Taxonomy Induction from Heterogenous Evidence. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 801–808, Sydney, Australia. Association for Computational Linguistics.
- Erik Tjong Kim Sang and Katja Hofmann. 2009. Lexical Patterns or Dependency Patterns: Which Is Better for Hypernym Extraction? In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 174–182, Boulder, Colorado, USA. Association for Computational Linguistics.
- Ivan Vulić and Anna Korhonen. 2016. On the Role of Seed Lexicons in Learning Bilingual Word Embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 247–257, Berlin, Germany. Association for Computational Linguistics.
- Ekaterina Vylomova, Laura Rimell, Trevor Cohn, and Timothy Baldwin. 2016. Take and Took, Gaggle and Goose, Book and Read: Evaluating the Utility of Vector Differences for Lexical Relation Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1671–1682, Berlin, Germany. Association for Computational Linguistics.
- Tonio Wandmacher. 2005. How semantic is Latent Semantic Analysis? In *Proceedings of RÉCITAL 2005*, pages 525–534, Dourdan, France.
- Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir, and Bill Keller. 2014. Learning to Distinguish Hyponyms and Co-Hyponyms. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2249–2259, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Josuke Yamane, Tomoya Takatani, Hitoshi Yamada, Makoto Miwa, and Yutaka Sasaki. 2016. Distributional Hypernym Generation by Jointly Learning Clusters and Projections. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1871–1879, Osaka, Japan, December. The COLING 2016 Organizing Committee.

Guangyou Zhou, Yang Liu, Fang Liu, Daojian Zeng, and Jun Zhao. 2013. Improving Question Retrieval in Community Question Answering Using World Knowledge. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, IJCAI ’13*, pages 2239–2245, Beijing, China. AAAI Press.

A.5 Paper “Every child should have parents: a taxonomy refinement algorithm based on hyperbolic term embeddings”

R. Aly, S. Acharya, A. Ossa, A. Köhn, C. Biemann, and A. Panchenko, “**Every Child Should Have Parents: A Taxonomy Refinement Algorithm Based on Hyperbolic Term Embeddings**,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 4811–4817, Association for Computational Linguistics, July 2019

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/P19-1474>

Every child should have parents: a taxonomy refinement algorithm based on hyperbolic term embeddings

Rami Aly¹, Shantanu Acharya², Alexander Ossa¹, Arne Köhn^{4,1},
Chris Biemann¹, and Alexander Panchenko^{3,1}

¹Universität Hamburg, Hamburg, Germany

²National Institute of Technology Mizoram, Aizawl, India

³Skolkovo Institute of Science and Technology, Moscow, Russia

⁴Saarland University, Saarbrücken, Germany

{Saly,2ossa,koehn,biemann,panchenko}@informatik.uni-hamburg.de

Abstract

We introduce the use of Poincaré embeddings to improve existing state-of-the-art approaches to domain-specific taxonomy induction from text as a signal for both relocating wrong hyponym terms within a (pre-induced) taxonomy as well as for attaching disconnected terms in a taxonomy. This method substantially improves previous state-of-the-art results on the SemEval-2016 Task 13 on taxonomy extraction. We demonstrate the superiority of Poincaré embeddings over distributional semantic representations, supporting the hypothesis that they can better capture hierarchical lexical-semantic relationships than embeddings in the Euclidean space.

1 Introduction

The task of taxonomy induction aims at creating a semantic hierarchy of entities by using hyponym-hypernym relations – called *taxonomy* – from text corpora. Compared to many other domains of natural language processing that make use of pre-trained dense representations, state-of-the-art taxonomy learning is still highly relying on traditional approaches like extraction of lexical-syntactic patterns (Hearst, 1992) or co-occurrence information (Grefenstette, 2015). Despite the success of pattern-based approaches, most taxonomy induction systems suffer from a significant number of disconnected terms, since the extracted relationships are too specific to cover most words (Wang et al., 2017; Bordea et al., 2016). The use of distributional semantics for hypernym identification and relation representation has thus received increasing attention (Shwartz et al., 2016). However, Levy et al. (2015) observe that many proposed supervised approaches instead learn prototypical hypernyms (that are hypernyms to many

other terms), not taking into account the relation between both terms in classification. Therefore, past applications of distributional semantics appear to be rather unsuitable to be directly applied to taxonomy induction as the sole signal (Tan et al., 2015; Pocostales, 2016). We address that issue by introducing a series of simple and parameter-free refinement steps that employ word embeddings in order to improve existing domain-specific taxonomies, induced from text using traditional approaches in an unsupervised fashion.

We compare two types of dense vector embeddings: the standard word2vec CBOW model (Mikolov et al., 2013a,b), that embeds terms in *Euclidean space* based on distributional similarity, and the more recent Poincaré embeddings (Nickel and Kiela, 2017), which capture similarity as well as hierarchical relationships in a *hyperbolic space*. The source code has been published¹ to recreate the employed embedding, to refine taxonomies as well as to enable further research of Poincaré embeddings for other semantic tasks.

2 Related Work

The extraction of taxonomic relationships from text corpora is a long-standing problem in ontology learning, see Biemann (2005) for an earlier survey. Wang et al. (2017) discuss recent advancements in taxonomy construction from text corpora. Conclusions from the survey include: i) The performance of extraction of IS-A relation can be improved by studying how pattern-based and distributional approaches complement each other; ii) there is only limited success of pure deep learn-

¹https://github.com/uhh-lt/Taxonomy_Refinement_EMBEDDINGS

ing paradigms here, mostly because it is difficult to design a single objective function for this task.

On the two recent TExEval tasks at SemEval for taxonomy extraction (Bordea et al., 2015, 2016), attracting a total of 10 participating teams, attempts to primarily use a distributional representation failed. This might seem counterintuitive, as taxonomies are surely modeling semantics and thus their extraction should benefit from semantic representations. The 2015 winner INRIASAC (Grefenstette, 2015) performed relation discovery using substring inclusion, lexical-syntactic patterns and co-occurrence information based on sentences and documents from Wikipedia. The winner in 2016, TAXI (Panchenko et al., 2016), harvests hypernyms with substring inclusion and Hearst-style lexical-syntactic patterns (Hearst, 1992) from domain-specific texts obtained via focused web crawling. The only submission to the TExEval 2016 task that relied exclusively on distributional semantics to induce hypernyms by adding a vector offset to the corresponding hyponym (Pocostales, 2016) achieved only modest results. A more refined approach to applying distributional semantics by Zhang et al. (2018) generates a hierarchical clustering of terms with each node consisting of several terms. They find concepts that should stay in the same cluster using embedding similarity – whereas, similar to the TExEval task, we are interested in making distinctions between all terms. Finally, Le et al. (2019) also explore using Poincaré embeddings for taxonomy induction, evaluating their method on hypernymy detection and reconstructing WordNet. However, in contrast to our approach that filters and attaches terms, they perform inference.

3 Taxonomy Refinement using Hyperbolic Word Embeddings

We employ embeddings using distributional semantics (i.e. word2vec CBOW) and Poincaré embeddings (Nickel and Kiela, 2017) to alleviate the largest error classes in taxonomy extraction: the existence of *orphans* – disconnected nodes that have an overall connectivity degree of zero and *outliers* – a child node that is assigned to a wrong parent. The rare case in which multiple parents can be assigned to a node has been ignored in the proposed refinement system. The first step consists of creating domain-specific Poincaré embeddings (§ 3.1). They are then used to identify and

relocate outlier terms in the taxonomy (§ 3.2), as well as to attach unconnected terms to the taxonomy (§ 3.3). In the last step, we further optimize the taxonomy by employing the endocentric nature of hypernyms (§ 3.4). See Figure 1 for a schematic visualization of the refinement pipeline. In our experiments, we use the output of three different systems. The refinement method is generically applicable to (noisy) taxonomies, yielding an improved taxonomy extraction system overall.

3.1 Domain-specific Poincaré Embedding

Training Dataset Construction To create domain-specific Poincaré embeddings, we use noisy hypernym relationships extracted from a combination of general and domain-specific corpora. For the general domain, we extracted 59.2 GB of text from English Wikipedia, Gigaword (Parker et al., 2009), ukWac (Ferraresi et al., 2008) and LCC news corpora (Goldhahn et al., 2012). The domain-specific corpora consist of web pages, selected by using a combination of BootCat (Baroni and Bernardini, 2004) and focused crawling (Remus and Biemann, 2016). Noisy IS-A relations are extracted with lexical-syntactic patterns from all corpora by applying PattaMaika², PatternSim (Panchenko et al., 2012), and WebISA (Seitner et al., 2016), following (Panchenko et al., 2016).³

The extracted noisy relationships of the common and domain-specific corpora are further processed separately and combined afterward. To limit the number of terms and relationships, we restrict the IS-A relationships on pairs for which both entities are part of the taxonomy’s vocabulary. Relations with a frequency of less than three are removed to filter noise. Besides further removing every reflexive relationship, only the more frequent pair of a symmetric relationship is kept. Hence, the set of cleaned relationships is transformed into being antisymmetric and irreflexive. The same procedure is applied to relationships extracted from the general-domain corpus with a frequency cut-off of five. They are then used to expand the set of relationships created from the domain-specific corpora.

²<http://jobimtext.org>: The PattaMaika component is based on UIMA RUTA (Kluegl et al., 2016).

³Alternatively to the relations extracted using lexical patterns, we also tried to use hypernyms extracted using the pre-trained HypeNet model (Shwartz et al., 2016), but the overall taxonomy evaluation results were lower than the standard baseline of the TAXI system and thus are not presented here.

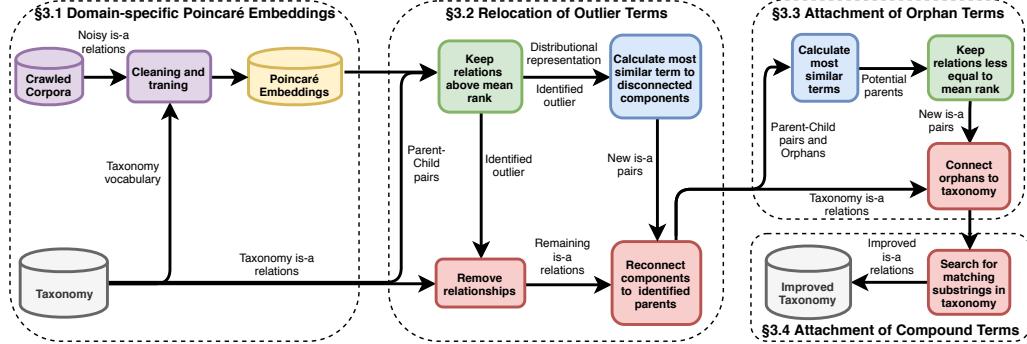


Figure 1: Outline of our taxonomy refinement method, with paper sections indicated.

Hypernym-Hyponym Distance Poincaré embeddings are trained on these cleaned IS-A relationships. For comparison, we also trained a model on noun pairs extracted from WordNet (PN). Pairs were only kept if both nouns were present in the vocabulary of the taxonomy. Finally, we trained the word2vec embeddings, connecting compound terms in the training corpus (Wikipedia) by ‘_’ to learn representations for compound terms, i.e multiword units, for the input vocabulary.

In contrast to embeddings in the Euclidean space where the cosine similarity $\frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \|\mathbf{v}\|}$ is commonly applied as a similarity measure, Poincaré embeddings use a hyperbolic space, specifically the Poincaré ball model (Stillwell, 1996). Hyperbolic embeddings are designed for modeling hierarchical relationships between words as they explicitly capture the hierarchy between words in the embedding space and are therefore a natural fit for inducing taxonomies. They were also successfully applied to hierarchical relations in image classification tasks (Khrulkov et al., 2019). The distance between two points $\mathbf{u}, \mathbf{v} \in \mathcal{B}^d$ for a d -dimensional Poincaré Ball model is defined as:

$$d(\mathbf{u}, \mathbf{v}) = \text{arcosh} \left(1 + 2 \frac{\|\mathbf{u} - \mathbf{v}\|^2}{(1 - \|\mathbf{u}\|^2)(1 - \|\mathbf{v}\|^2)} \right).$$

This Poincaré distance enables us to capture the hierarchy and similarity between words simultaneously. It increases exponentially with the depth of the hierarchy. So while the distance of a leaf node to most other nodes in the hierarchy is very high, nodes on abstract levels, such as the root, have a comparably small distance to all nodes in the hierarchy. The word2vec embeddings have no notion of hierarchy and hierarchical relationships cannot be represented with vector offsets across

the vocabulary (Fu et al., 2014). When applying word2vec, we use the observation that distributionally similar words are often co-hyponyms (Heylen et al., 2008; Weeds et al., 2014).

3.2 Relocation of Outlier Terms

Poincaré embeddings are used to compute and store a rank $\text{rank}(x, y)$ between every child and parent of the existing taxonomy, defined as the index of y in the list of sorted Poincaré distances of all entities of the taxonomy to x . Hypernym-hyponym relationships with a rank larger than the mean of all ranks are removed, chosen on the basis of tests on the 2015 TExEval data (Bordea et al., 2015). Disconnected components that have children are re-connected to the most similar parent in the taxonomy or to the taxonomy root if no distributed representation exists. Previously or now disconnected isolated nodes are subject to orphan attachment (§ 3.3).

Since distributional similarity does not capture parent-child relations, the relationships are not registered as parent-child but as co-hyponym relationships. Thus, we compute the distance to the closest co-hyponym (child of the same parent) for every node. This filtering technique is then applied to identify and relocate outliers.

3.3 Attachment of Orphan Terms

We then attach orphans (nodes unattached in the input or due to the removal of relationships in the previous step) by computing the rank between every orphan and the most similar node in the taxonomy. This node is an orphan’s potential parent. Only hypernym-hyponym relationships with a rank lower or equal to the mean of all stored ranks are added to the taxonomy. For the word2vec system, a link is added between the parent of the most

similar co-hyponym and the orphan.

3.4 Attachment of Compound Terms

In case a representation for a compound noun term does not exist, we connect it to a term that is a substring of the compound. If no such term exists, the noun remains disconnected. Finally, the Tarjan algorithm (Tarjan, 1972) is applied to ensure that the refined taxonomy is asymmetric: In case a circle is detected, one of its links is removed at random.

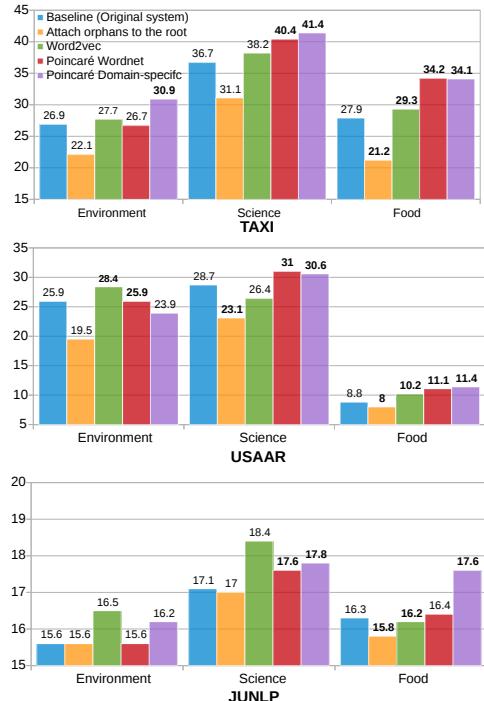


Figure 2: F_1 results for the systems on all domains. Vocabulary sizes: environment ($|V| = 261$), science ($|V| = 453$), food ($|V| = 1556$). **Bold** numbers are significantly different results to the original system with $p < 0.05$.

4 Evaluation

Proposed methods are evaluated on the data of SemEval2016 TExEval (Bordea et al., 2016) for submitted systems that created taxonomies for all domains of the task⁴, namely the task-winning system TAXI (Panchenko et al., 2016) as well as the systems USAAR (Tan et al., 2016) and JUNLP (Maitra and Das, 2016). TAXI harvests

⁴<http://alt.qcri.org/semeval2016/task13/index.php>

hypernyms with substring inclusion and lexical-syntactic patterns by obtaining domain-specific texts via focused web crawling. USAAR and JUNLP heavily rely on rule-based approaches. While USAAR exploits the endocentric nature of hyponyms, JUNLP combines two string inclusion heuristics with semantic relations from BabelNet. We use the taxonomies created by these systems as our baseline and additionally ensured that taxonomies do neither have circles nor in-going edges to the taxonomy root by applying the Tarjan algorithm (Tarjan, 1972), removing a random link from detected cycles. This causes slight differences between the baseline results in Figure 2 and (Bordea et al., 2016).

5 Results and Discussion

Comparison to Baselines Figure 2 shows comparative results for all datasets and measures for every system. The *Root* method, which connects all orphans to the root of the taxonomy, has the highest connectivity but falls behind in scores significantly. Word2vec CBOW embeddings partly increase the scores, however, the effect appears to be inconsistent. Word2vec embeddings connect more orphans to the taxonomy (cf. Table 2), albeit with mixed quality, thus the interpretation of word similarity as co-hyponymy does not seem to be appropriate. Word2vec as a means to detect hypernyms has shown to be rather unsuitable (Levy et al., 2015). Even more advanced methods such as the *diff* model (Fu et al., 2014) merely learn so-called *prototypical hypernyms*.

Both Poincaré embeddings variants outperform the word2vec ones yielding major improvements over the baseline taxonomy. Employing the McNemar (1947) significance test shows that Poincaré embeddings’ improvements to the original systems are indeed significant. The achieved improvements are larger on the TAXI system than on the other two systems. We attribute to the differences of these approaches: The rule-based approaches relying on string inclusion as carried out by USAAR and JUNLP are highly similar to step §3.4. Additionally, JUNLP creates taxonomies with many but very noisy relationships, therefore step §3.3 does not yield significant gains, since there are much fewer orphans available to connect to the taxonomy. This problem also affects the USAAR system for the food domain. For the environment domain, however, USAAR creates a

Word	Parent in TAXI	Parent after refinement	Gold parent	Closest neighbors
second language acquisition	—	linguistics	linguistics	applied linguistics, semantics, linguistics
botany	—	genetics	plant science, ecology	genetics, evolutionary ecology, animal science
sweet potatoes	—	vegetables	vegetables	vegetables, side dishes, fruit
wastewater	water	waste	waste	marine pollution, waste, pollutant
water	waste, natural resources	natural resources	aquatic environment	continental shelf, management of resources
international relations	sociology, analysis, humanities	humanities	political science	economics, economic theory, geography

Table 1: Example words with respective parent(s) in the input taxonomy and after refinement using our domain-specific Poincaré embeddings, as well as the word’s closest three neighbors (incl. orphans) in embeddings.

Domain	word2vec	P. WordNet	P. domain-specific	# orphans
Environment	25	18	34	113
Science	56	39	48	158
Food	347	181	267	775

Table 2: Number of attached orphans in taxonomies created by TAXI using different embeddings.

taxonomy with very high precision but low recall which makes step §3.2 relatively ineffective. As step §3.3 has shown to improve scores more than §3.2, the gains on JUNLP are comparably lower.

WordNet-based Embeddings The domain-specific Poincaré embeddings mostly perform either comparably or outperform the WordNet-based ones. In error analysis, we found that while WordNet-based embeddings are more accurate, they have a lower coverage as seen in Table 2, especially for attaching complex multiword orphan vocabulary entries that are not contained in WordNet, e.g., *second language acquisition*. Based on the results we achieved by using domain-specific Poincaré embeddings, we hypothesize that their attributes result in a system that learns hierarchical relations between a pair of terms. The closest neighbors of terms in the embedding clearly tend to be more generic as exemplarily shown in Table 1, which further supports our claim. Their use also enables the correction of false relations created by string inclusion heuristics as seen with *wastewater*. However, we also notice that few and inaccurate relations for some words results in imprecise word representations such as for *botany*.

Multilingual Results Applying domain-specific Poincaré embeddings to other languages also creates overall improved taxonomies, however the scores vary as seen in Table 3. While the score of all food taxonomies increased substantially, the taxonomies quality for environment did not improve, it even declines. This seems to be due to the lack of extracted relations in (§3.1), which results in imprecise representations and a highly limited

vocabulary in the Poincaré embedding model, especially for Italian and Dutch. In these cases, the refinement is mostly defined by step §3.4.

Language	Domain	Original	Refined	# rel. data	# rel. gold
English	Environment	26.9	30.9	657	261
	Science	36.7	41.4	451	465
	Food	27.9	34.1	1898	1587
French	Environment	23.7	28.3	114	266
	Science	31.8	33.1	118	451
	Food	22.4	28.9	598	1441
Italian	Environment	31.0	30.8	2	266
	Science	32.0	34.2	4	444
	Food	16.9	18.5	57	1304
Dutch	Environment	28.4	27.1	7	267
	Science	29.8	30.5	15	449
	Food	19.4	21.8	61	1446

Table 3: F₁ comparison between original (TAXI) and refined taxonomy using domain-specific embeddings.

6 Conclusion

We presented a refinement method for improving existing taxonomies through the use of hyperbolic Poincaré embeddings. They consistently yield improvements over strong baselines and in comparison to word2vec as a representative for distributional vectors in the Euclidean space. We further showed that Poincaré embeddings can be efficiently created for a specific domain from crawled text without the need for an existing database such as WordNet. This observation confirms the theoretical capability of Poincaré embeddings to learn hierarchical relations, which enables their future use in a wide range of semantic tasks. A prominent direction for future work is using the hyperbolic embeddings as the sole signal for taxonomy extraction. Since distributional and hyperbolic embeddings cover different relations between terms, it may be interesting to combine them.

Acknowledgments

We acknowledge the support of DFG under the “JOIN-T” (BI 1544/4) and “ACQuA” (BI 1544/7) projects as well as the DAAD. We also thank three anonymous reviewers and Simone Paolo Ponzerotto for providing useful feedback on this work.

References

- Marco Baroni and Silvia Bernardini. 2004. **Bootcat: Bootstrapping corpora and terms from the web.** In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, pages 1313–1316, Lisbon, Portugal.
- Chris Biemann. 2005. Ontology learning from text: A survey of methods. *LDV Forum*, 20(2):75–93.
- Georgeta Bordea, Paul Buitelaar, Stefano Faralli, and Roberto Navigli. 2015. **Semeval-2015 task 17: Taxonomy Extraction Evaluation (TExEval).** In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 902–910, Denver, CO, USA.
- Georgeta Bordea, Els Lefever, and Paul Buitelaar. 2016. **SemEval-2016 Task 13: Taxonomy Extraction Evaluation (TExEval-2).** In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1081–1091, San Diego, CA, USA.
- Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. **Introducing and evaluating ukWaC, a very large web-derived corpus of English.** In *Proceedings of the 4th Web as Corpus Workshop. Can we beat Google?*, pages 47–54, Marrakech, Morocco.
- Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. **Learning semantic hierarchies via word embeddings.** In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 1199–1209, Baltimore, MD, USA.
- Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. **Building large monolingual dictionaries at the Leipzig Corpora Collection: From 100 to 200 languages.** In *Proceedings of the Eight International Conference on Language Resources and Evaluation*, pages 759–765, Istanbul, Turkey.
- Gregory Grefenstette. 2015. **INRIASAC: Simple Hypernym Extraction Methods.** In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 911–914, Denver, CO, USA.
- Marti A. Hearst. 1992. **Automatic acquisition of hyponyms from large text corpora.** In *Proceedings of the 14th Conference on Computational Linguistics*, pages 539–545, Nantes, France.
- Kris Heylen, Yves Peirsman, Dirk Geeraerts, and Dirk Speelman. 2008. **Modelling word similarity: an evaluation of automatic synonymy extraction algorithms.** In *Proceedings of the sixth international language resources and evaluation*, pages 3243–3249, Marrakech, Morocco.
- Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. 2019. **Hyperbolic image embeddings.** In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Peter Kluegl, Martin Toepper, Philip-Daniel Beck, Georg Fette, and Frank Puppe. 2016. **UIMA Ruta: Rapid development of rule-based information extraction applications.** *Natural Language Engineering*, 22(1):1–40.
- Matt Le, Stephen Roller, Laetitia Papaxanthos, Douwe Kiela, and Maximilian Nickel. 2019. **Inferring concept hierarchies from text corpora via hyperbolic embeddings.** *arXiv preprint arXiv:1902.00913*.
- Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. 2015. **Do supervised distributional methods really learn lexical inference relations?** In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 970–976, Denver, CO, USA.
- Promita Maitra and Dipankar Das. 2016. **JUNLP at SemEval-2016 task 13: A language independent approach for hypernym identification.** In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1310–1314, San Diego, CA, USA.
- Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.
- Tomas Mikolov, Kai Chen, G.S Corrado, and Jeffrey Dean. 2013a. **Efficient estimation of word representations in vector space.** In *ICLR Workshop*, Scottsdale, AZ, USA.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. **Distributed representations of words and phrases and their compositionality.** In *Advances in neural information processing systems*, pages 3111–3119, Stateline, NV, USA.
- Maximillian Nickel and Douwe Kiela. 2017. **Poincaré embeddings for learning hierarchical representations.** In *Advances in Neural Information Processing Systems 30*, pages 6338–6347, Long Beach, CA, USA.
- Alexander Panchenko, Stefano Faralli, Eugen Ruppert, Steffen Remus, Hubert Naets, Cédric Fairon, Simone P. Ponzetto, and Chris Biemann. 2016. **TAXI at SemEval-2016 Task 13: a taxonomy Induction Method based on Lexico-syntactic Patterns, Substrings and Focused Crawling.** In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1320–1327, San Diego, CA, USA.
- Alexander Panchenko, Olga Morozova, and Hubert Naets. 2012. **A semantic similarity measure based on lexico-syntactic patterns.** In *Proceedings of KONVENTS 2012*, pages 174–178, Vienna, Austria.

- Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2009. English gigaword forth edition. In *Linguistic Data Consortium*, Philadelphia, PA, USA.
- Joel Pocostales. 2016. **NUIG-UNLP at SemEval-2016 Task 13: A Simple Word Embedding-based Approach for Taxonomy Extraction**. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1298–1302, San Diego, CA, USA.
- Steffen Remus and Chris Biemann. 2016. **Domain-specific corpus expansion with focused webcrawling**. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, pages 3607–3611, Portorož, Slovenia.
- Julian Seitner, Christian Bizer, Kai Eckert, Stefano Faralli, Robert Meusel und Heiko Paulheim, and Simone P. Ponzetto. 2016. **A large database of hypernymy relations extracted from the Web**. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, pages 360–367, Portorož, Slovenia.
- Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. **Improving hypernymy detection with an integrated path-based and distributional method**. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 2389–2398, Berlin, Germany.
- John Stillwell. 1996. *Sources of hyperbolic geometry*. History of Mathematics, Volume 10. American Mathematical Society.
- Liling Tan, Francis Bond, and Josef van Genabith. 2016. **USAAR at SemEval-2016 task 13: Hyponym endocentricity**. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1303–1309, San Diego, CA, USA.
- Liling Tan, Rohit Gupta, and Josef van Genabith. 2015. **USAAR-WLV: Hypernym generation with Deep Neural Nets**. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 932–937, Denver, CO, USA.
- Robert Tarjan. 1972. Depth first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160.
- Chengyu Wang, Xiaofeng He, and Aoying Zhou. 2017. **A short survey on taxonomy learning from text corpora: Issues, resources and recent advances**. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1190–1203, Copenhagen, Denmark.
- Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir, and Bill Keller. 2014. **Learning to distinguish hypernyms and co-hypernyms**. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics*, pages 2249–2259, Dublin, Ireland.
- Chao Zhang, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian Sadler, Michelle Vanni, and Jiawei Han. 2018. **TaxoGen: Unsupervised topic taxonomy construction by adaptive term embedding and clustering**. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2701–2709, London, United Kingdom.

A.6 Article “Watset: Local-Global Graph Clustering with Applications in Sense and Frame Induction”

D. Ustalov, A. Panchenko, C. Biemann, and S. P. Ponzetto, “Watset: Local-Global Graph Clustering with Applications in Sense and Frame Induction,” *Computational Linguistics*, vol. 45, pp. 423–479, Sept. 2019

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/J19-3002>

WATSET: Local-Global Graph Clustering with Applications in Sense and Frame Induction

Dmitry Ustalov

Data & Web Science Group

University of Mannheim

dmitry@informatik.uni-mannheim.de

Alexander Panchenko

Language Technology Group,

Skolkovo Institute of Science

and Technology

panchenko@informatik.

uni-hamburg.de

Chris Biemann

Language Technology Group

University of Hamburg

biemann@informatik.uni-hamburg.de

Simone Paolo Ponzetto

Data & Web Science Group

University of Mannheim

simone@informatik.uni-mannheim.de

We present a detailed theoretical and computational analysis of the Watset meta-algorithm for fuzzy graph clustering, which has been found to be widely applicable in a variety of domains. This algorithm creates an intermediate representation of the input graph, which reflects the “ambiguity” of its nodes. Then, it uses hard clustering to discover clusters in this “disambiguated” intermediate graph. After outlining the approach and analyzing its computational complexity, we demonstrate that Watset shows competitive results in three applications: unsupervised synset induction from a synonymy graph, unsupervised semantic frame induction from dependency triples, and unsupervised semantic class induction from a distributional thesaurus. Our algorithm is generic and can also be applied to other networks of linguistic data.

1. Introduction

Language can be conceived as a system of interrelated symbols, such as words, senses, part-of-speeches, letters, and so forth. Ambiguity is a fundamental inherent property

Submission received: 20 August 2018; revised version received: 22 February 2019; accepted for publication: 15 March 2019.

https://doi.org/10.1162/COLI_a_00354

© 2019 Association for Computational Linguistics
Published under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International
(CC BY-NC-ND 4.0) license

of language. Namely, each symbol can refer to several meanings, mapping the space of objects to the space of communicative signs (de Saussure 1916). For language processing applications, these symbols need to be represented in a computational format. The structure discovery paradigm (Biemann 2012) aims at *inducing a system of linguistic symbols and relationships between them* in an unsupervised way to enable processing of a wide variety of languages. Clustering algorithms are central and ubiquitous tools for such kinds of unsupervised structure discovery processes applied to natural language data. In this article, we present a new clustering algorithm,¹ which is especially suitable for processing graphs of linguistic data, because it performs disambiguation of symbols in the local context in order to subsequently globally cluster those disambiguated symbols.

At the heart of our method lies the pre-processing of a graph on the basis of local pre-clustering. Breaking nodes that connect to several communities (i.e., hubs) into several local senses helps to better reach the goal of clustering, no matter which clustering algorithm is used. This results in a sparser sense-aware graphical representation of the input data. Such a representation allows the use of efficient hard clustering algorithms for performing fuzzy clustering.

The contributions presented in this article include:

1. A **meta-algorithm for graph clustering**, called WATSET, performing a fuzzy clustering of the input graph using hard clustering methods in two subsequent steps (Section 3).
2. A **method for synset induction** based on the WATSET algorithm applied to synonymy graphs weighted by word embeddings (Section 4).
3. A **method for semantic frame induction** based on the WATSET algorithm applied as a triclustering algorithm to syntactic triples (Section 5).
4. A **method for semantic class induction** based on the WATSET algorithm applied to a distributional thesaurus (Section 6).

This article is organized as follows. Section 2 discusses the related work. Section 3 presents the WATSET algorithm in a more general fashion than previously introduced in Ustalov, Panchenko, and Biemann (2017), including an analysis of its computational complexity and run-time. We also describe a simplified version of WATSET that does not use the context similarity measure for propagating links in the original graph to the appropriate senses in the disambiguated graph. Three subsequent sections present different applications of the algorithm. Section 4 applies WATSET for unsupervised synset induction, referencing results by Ustalov, Panchenko, and Biemann. Section 5 shows frame induction with WATSET on the basis of a triclustering approach, as previously described by Ustalov et al. (2018). Section 6 presents new experiments on semantic class induction with WATSET. Section 7 concludes with the final remarks and pointers for future work.

Table 1 shows several examples of linguistic structures on which we conduct experiments described in this article. With the exception of the type of input graph and the hyper-parameters of the WATSET algorithm, the overall pipeline remains similar in every described application. For instance, in Section 4 the input of the clustering algorithm is a graph of ambiguous synonyms and the output is an induced linguistic

¹ This article builds upon and expands on Ustalov, Panchenko, and Biemann (2017) and Ustalov et al. (2018).

Table 1

Various types of input linguistic graphs clustered by the WATSET algorithm and the corresponding induced output symbolic linguistic structures.

Input Nodes	Input Edges	Output Linguistic Structure	See
Polysemous words	Synonymy relationships	Synsets composed of disambiguated words	§ 4
Subject-Verb-Object (SVO) triples	Most distributionally similar SVO triples	Lexical semantic frames	§ 5
Polysemous words	Most distributionally similar words	Semantic classes composed of disambiguated words	§ 6

structure that represents synsets. Thus, by varying the input graphs we show how using the same methodology on various types of linguistic structures can be induced in an unsupervised manner. This opens avenues for extraction of various meaningful structures from linguistic graphs in natural language processing (NLP) and other fields using the method presented in this article.

2. Related Work

We present surveys on graph clustering (Section 2.1), word sense induction (Section 2.2), lexical semantic frame induction (Section 2.3), and semantic class induction (Section 2.4), giving detailed explanations of algorithms used in our experiments and discussing related work on these topics.

2.1 Graph Clustering

Graph clustering is a process of finding groups of strongly related vertices in a graph, which is a field of research in its own right with a large number of proposed approaches; see Schaeffer (2007) for a survey. Graph clustering methods are strongly related to the methods for finding communities in networks (Newman and Girvan 2004; Fortunato 2010). In our work, we focus mostly on the algorithms, which have proven to be useful for processing of networks of *linguistic data*, such as word co-occurrence graphs, especially those that were used for induction of linguistic structures such as word senses.

Markov Clustering (MCL; van Dongen 2000) is a *hard* clustering algorithm, that is, a method that partitions nodes of the graph in a set of disjoint clusters. This method is based on simulation of stochastic flow in graphs. MCL simulates random walks within a graph by the alternation of two operators, called expansion and inflation, which recompute the class labels. Notably, it has been successfully used for the word sense induction task (Dorow and Widdows 2003).

Chinese Whispers (CW; Biemann 2006, 2012) is a *hard* clustering algorithm for weighted graphs, which can be considered as a special case of MCL with a simplified class update step. At each iteration, the labels of all the nodes are updated according to the majority of labels among the neighboring nodes. The algorithm has a hyperparameter that controls graph weights, which can be set to three values: (1) CW_{top} sums

over the neighborhood's classes; (2) CW_{lin} downgrades the influence of a neighboring node by its degree; or (3) CW_{log} by the logarithm of its degree.

MaxMax (Hope and Keller 2013a) is a *fuzzy* clustering algorithm particularly designed for the word sense induction task. In a nutshell, pairs of nodes are grouped if they have a maximal mutual affinity. The algorithm starts by converting the undirected input graph into a directed graph by keeping the maximal affinity nodes of each node. Next, all nodes are marked as root nodes. Finally, for each root node, the following procedure is repeated: All transitive children of this root form a cluster and the roots are marked as non-root nodes; a root node together with all its transitive children form a fuzzy cluster.

Clique Percolation Method (CPM) by Palla et al. (2005) is a *fuzzy* clustering algorithm, that is, a method that partitions nodes of a graph in a set of potentially overlapping clusters. The method is designed for unweighted graphs and builds up clusters from k -cliques corresponding to fully connected sub-graphs of k nodes. Although this method is only commonly used in social network analysis for clique detection, we decided to add it to the comparison, as synsets are essentially cliques of synonyms.

The **Louvain** method (Blondel et al. 2008) is a *hard* graph clustering method developed for identification of communities in large networks. The algorithm finds hierarchies of clusters in a recursive fashion. It is based on a greedy method that optimizes modularity of a partition of the network. First, it looks for small communities by optimizing modularity locally. Second, it aggregates nodes belonging to the same community and builds a new network whose nodes are the communities. These steps are repeated to maximize modularity of the clustering result.

2.2 Word Sense Induction

Word Sense Induction is an unsupervised knowledge-free approach to Word Sense Disambiguation (WSD): It uses neither handcrafted lexical resources nor hand-annotated sense-labeled corpora. Instead, it induces word sense inventories automatically from corpora. Unsupervised WSD methods fall into two main categories: context clustering and word ego network clustering.

Context clustering approaches, such as Pedersen and Bruce (1997) and Schütze (1998), represent an instance usually by a vector that characterizes its context, where the definition of context can vary greatly. These vectors of each instance are then clustered.

Schütze (1998) induced sparse sense vectors by clustering context vectors, using the expectation-maximization algorithm. This approach is fitted with a similarity-based WSD mechanism. Pantel and Lin (2002) used a two-staged *Clustering by Committee* algorithm. In the first stage, it uses average-link clustering to find small and tight clusters, which are used to iteratively identify committees from these clusters. Reisinger and Mooney (2010) presented a multi-prototype vector space. Sparse tf-idf vectors are clustered, using a parametric method fixing the same number of senses for all words. Sense vectors are centroids of the clusters.

Whereas most dense word vector models represent a word with a single vector and thus conflate senses (Mikolov et al. 2013; Pennington, Socher, and Manning 2014), there are several approaches that produce word sense embeddings. Multi-prototype extensions of the Skip-Gram model (Mikolov et al. 2013) that use no predefined sense inventory learn one embedding word vector per one word sense and are commonly fitted with a disambiguation mechanism (Huang et al. 2012; Apidianaki and Sagot 2014;

Neelakantan et al. 2014; Tian et al. 2014; Li and Jurafsky 2015; Bartunov et al. 2016; Cocos and Callison-Burch 2016; Pelevina et al. 2016; Thomason and Mooney 2017).

Huang et al. (2012) introduced multiple word prototypes for dense vector representations (embeddings). Their approach is based on a neural network architecture; during training, all contexts of the word are clustered.

Apidianaki and Sagot (2014) use an aligned parallel corpus and WordNet for English to perform cross-lingual word sense disambiguation to produce French synsets. However, Cocos and Callison-Burch (2016) showed that it is possible to successfully perform a monolingual word sense induction using only such a paraphrase corpus as Paraphrase Database (Pavlick et al. 2015).

Tian et al. (2014) introduced a probabilistic extension of the Skip-Gram model (Mikolov et al. 2013) that learns multiple sense-aware prototypes weighted by their prior probability. These models use parametric clustering algorithms that produce a fixed number of senses per word.

Neelakantan et al. (2014) proposed a multi-sense extension of the Skip-Gram model, which was the first one to learn the number of senses by itself. During training, a new sense vector is allocated if the current context's similarity to existing senses is below some threshold. All previously mentioned sense embeddings were evaluated on the contextual word similarity task, each one improving upon previous models.

Nieto Piña and Johansson (2015) presented another multi-prototype modification of the Skip-Gram model. Their approach outperforms that of Neelakantan et al. (2014), but requires the number of senses for each word to be set manually.

Bartunov et al. (2016) introduced AdaGram, a non-parametric method for learning sense embeddings based on a Bayesian extension of the Skip-Gram model. The granularity of learned sense embeddings is controlled by the α parameter.

Li and Jurafsky (2015) proposed an approach for learning sense embeddings based on the Chinese Restaurant Process. A new sense is allocated if a new word context is significantly different from existing senses. The approach was tested on multiple NLP tasks, showing that sense embeddings can significantly improve the performance of part-of-speech tagging, semantic relationship identification, and semantic relatedness tasks, but yield no improvement for named entity recognition and sentiment analysis.

Thomason and Mooney (2017) performed multi-modal word sense induction by combining both language and vision signals. In this approach, word embeddings are learned from the ImageNet corpus (Deng et al. 2009) and visual features are obtained from a deep neural network. Running a k -means algorithm on the joint feature set produces WordNet-like synsets.

Word ego network clustering methods cluster graphs of words semantically related to the ambiguous word (Lin 1998; Pantel and Lin 2002; Widdows and Dorow 2002; Biemann 2006; Hope and Keller 2013a). An ego network consists of a single node (ego), together with the nodes they are connected to (alters), and all the edges among those alters (Everett and Borgatti 2005). In our case, such a network is a local neighborhood of one word. Nodes of the ego network can be (1) words semantically similar to the target word, as in our approach, or (2) context words relevant to the target, as in the *UoS* system (Hope and Keller 2013b). Graph edges represent semantic relationships between words derived using corpus-based methods (e.g., distributional semantics) or gathered from dictionaries. The sense induction process using word graphs is explored by Widdows and Dorow (2002), Biemann (2006), and Hope and Keller (2013a). Disambiguation of instances is performed by assigning the sense with the highest overlap between the instance's context words and the words of the sense cluster. Véronis (2004) compiles a corpus with contexts of polysemous nouns using a search engine.

A word graph is built by drawing edges between co-occurring words in the gathered corpus, where edges below a certain similarity threshold were discarded. His HyperLex algorithm detects hubs of this graph, which are interpreted as word senses. Disambiguation in this experiment is performed by computing the distance between context words and hubs in this graph.

Di Marco and Navigli (2013) present a comprehensive study of several graph-based WSI methods, including CW, HyperLex, and curvature clustering (Dorow et al. 2005). Additionally, the authors propose two novel algorithms: Balanced Maximum Spanning Tree Clustering and Squares (B-MST), and Triangles and Diamonds (SquaT++). To construct graphs, authors use first-order and second-order relationships extracted from a background corpus as well as keywords from snippets. This research goes beyond intrinsic evaluations of induced senses and measures the impact of the WSI in the context of an information retrieval via clustering and diversifying Web search results. Depending on the data set, HyperLex, B-MST, or CW provided the best results. For a comparative study of graph clustering algorithms for word sense induction in a pseudo-word evaluation confirming the effectiveness of CW, see Cecchini et al. (2018).

Methods based on clustering of synonyms, such as our approach and Max-Max (Hope and Keller 2013a), induce the resource from an ambiguous graph of synonyms where edges are extracted from manually created resources. To the best of our knowledge, most experiments either used graph-based word sense induction applied to text-derived graphs or relied on a linking-based method that already assumes the availability of a WordNet-like resource. A notable exception is the ECO (Extraction, Clustering, Ontologization) approach by Gonçalo Oliveira and Gomes (2014), which was applied to induce a WordNet of the Portuguese language called Onto.PT.² ECO is a *fuzzy* clustering algorithm that was used to induce synsets for a Portuguese WordNet from several available synonymy dictionaries. The algorithm starts by adding random noise to edge weights. Then, the approach applies Markov Clustering (Section 2.1) of this graph several times to estimate the probability of each word pair being in the same synset. Finally, candidate pairs over a certain threshold are added to output synsets. We compare this approach to five other state-of-the-art graph clustering algorithms described in Section 2.1 as the baselines.

2.3 Semantic Frame Induction

Frame Semantics was originally introduced by Fillmore (1982) and further developed in the FrameNet project (Baker, Fillmore, and Lowe 1998). FrameNet is a lexical resource composed of a collection of semantic frames, relationships between them, and a corpus of frame occurrences in text. This annotated corpus gave rise to the development of frame parsers using supervised learning (Gildea and Jurafsky 2002; Erk and Padó 2006; Das et al. 2014, *inter alia*), as well as its application to a wide range of tasks, ranging from answer extraction in Question Answering (Shen and Lapata 2007) and Textual Entailment (Burchardt et al. 2009; Ben Aharon, Szpektor, and Dagan 2010).

However, frame-semantic resources are arguably expensive and time-consuming to build because of difficulties in defining the frames, their granularity and domain, as well as the complexity of the construction and annotation tasks. Consequently, such resources exist only for a few languages (Boas 2009) and even English is lacking domain-specific frame-based resources. Possible inroads are cross-lingual semantic annotation

² <http://ontopt.dei.uc.pt>.

transfer (Padó and Lapata 2009; Hartmann, Eckle-Kohler, and Gurevych 2016) or linking FrameNet to other lexical-semantic or ontological resources (Narayanan et al. 2003; Tonelli and Pighin 2009; Laparra and Rigau 2010; Gurevych et al. 2012, *inter alia*). One inroad for overcoming these issues is automatizing the process of FrameNet construction through unsupervised frame induction techniques, as investigated by the systems described next.

LDA-Frames (Materna 2012, 2013) is an approach to inducing semantic frames using a latent Dirichlet allocation (LDA) by Blei, Ng, and Jordan (2003) for generating semantic frames and their respective frame-specific semantic roles at the same time. The authors evaluated their approach against the CPA corpus (Hanks and Pustejovsky 2005). Although Ritter, Mausam, and Etzioni (2010) have applied LDA for inducing structures similar to frames, their study is focused on the extraction of mutually related frame arguments.

ProFinder (Cheung, Poon, and Vanderwende 2013) is another generative approach that also models both frames and roles as latent topics. The evaluation was performed on the in-domain information extraction task MUC-4 (Sundheim 1992) and on the text summarization task TAC-2010.³

Modi, Titov, and Klementiev (2012) build on top of an unsupervised semantic role labeling model (Titov and Klementiev 2012). The raw text of sentences from the FrameNet data is used for training. The FrameNet gold annotations are then used to evaluate the labeling of the obtained frames and roles, effectively clustering instances known during induction.

Kawahara, Peterson, and Palmer (2014) harvest a huge collection of verbal predicates along with their argument instances and then apply the Chinese Restaurant Process clustering algorithm to group predicates with similar arguments. The approach was evaluated on the verb cluster data set of Korhonen, Krymolowski, and Marx (2003).

These and some other related approaches (e.g., the one by O'Connor 2013), were all evaluated in completely different incomparable settings, and used different input corpora, making it difficult to judge their relative performance.

2.4 Semantic Class Induction

The problem of inducing semantic classes from text, also known as semantic lexicon induction, has also been extensively explored in previous works. This is because inducing semantic classes directly from text has the potential to avoid the limited coverage problems of knowledge bases like Freebase, DBpedia (Bizer et al. 2009), or BabelNet (Navigli and Ponzetto 2012), which rely on Wikipedia (Hovy, Navigli, and Ponzetto 2013), as well as to allow for resource induction across domains (Hovy et al. 2011). Information about semantic classes, in turn, has been shown to benefit such high-level NLP tasks as coreference (Ng 2007).

Induction of semantic classes as a research direction in the field of NLP starts, to the best of our knowledge, with Lin and Pantel (2001), where sets of similar words are clustered into concepts. This approach performs a hard clustering and does not label clusters, but these drawbacks are addressed by Pantel and Lin (2002), where words can belong to several clusters, thus representing senses.

Pantel and Ravichandran (2004) aggregate hypernyms per cluster, which come from Hearst (1992) patterns. Pattern-based approaches were further developed using

³ <https://tac.nist.gov/2010/Summarization>.

graph-based methods using a PageRank-based weighting (Kozareva, Riloff, and Hovy 2008), random walks (Talukdar et al. 2008), or heuristic scoring (Qadir et al. 2015). Other approaches use probabilistic graphical models, such as the ones proposed by Ritter, Mausam, and Etzioni (2010) and Hovy et al. (2011). To ensure the overall quality of extraction pattern with minimal supervision, Thelen and Riloff (2002) explored a bootstrapping approach, later extended by McIntosh and Curran (2009) with bagging and distributional similarity to minimize the semantic drift problem of iterative bootstrapping algorithms.

As an alternative to pattern-based methods, Pachenco et al. (2018b) show how to apply semantic classes to improve hypernymy extraction and taxonomy induction. Like in our experiments in Section 6, it uses a distributional thesaurus as input, as well as multiple pre- and post-processing stages to filter the input graph and disambiguate individual nodes. In contrast to Pachenco et al., here we directly apply the WATSET algorithm to obtain the resulting distributional semantic classes instead of using a sophisticated parametric pipeline that performs a sequence of clustering and pruning steps.

Another related strain of research to semantic class induction is dedicated to the automatic *set expansion* task (Sarmento et al. 2007; Wang and Cohen 2008; Pantel et al. 2009; Rong et al. 2016; Shen et al. 2017). In this task, a set of input lexical entries, such as words or entities, is provided (e.g., “apple, mango, pear, banana”). The system is expected to extend this initial set with relevant entries (such as other fruits in this case, e.g., “peach” and “lemon”). Besides the academic publications listed above, Google Sets was an industrial system for providing similar functionality.⁴

3. WATSET, an Algorithm for Fuzzy Graph Clustering

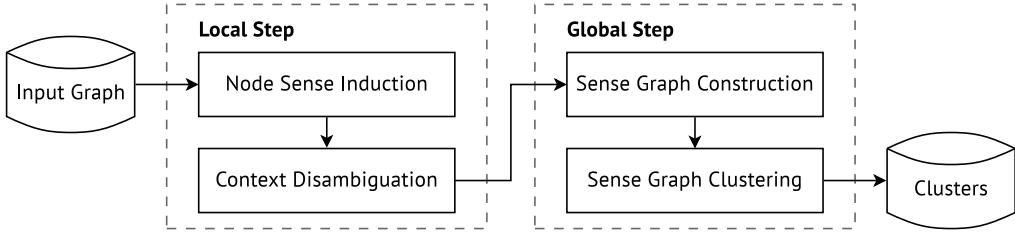
In this section, we present WATSET, a meta-algorithm for fuzzy graph clustering. Given a graph connecting potentially ambiguous objects (e.g., words), WATSET induces a set of unambiguous overlapping clusters (communities) by disambiguating and grouping the ambiguous objects. WATSET is a meta-algorithm that uses existing *hard* clustering algorithms for graphs to obtain a *fuzzy* clustering (e.g., *soft* clustering).

In computational linguistics, graph clustering is used for addressing problems such as word sense induction (Biemann 2006), lexical chain computing (Medelyan 2007), Web search results diversification (Di Marco and Navigli 2013), sentiment analysis (Pang and Lee 2004), and cross-lingual semantic relationship induction (Lewis and Steedman 2013b); more applications can be found in the book by Mihalcea and Radev (2011).

Definitions. Let $G = (V, E)$ be an undirected simple graph,⁵ where V is a set of nodes and $E \subseteq V^2$ is a set of undirected edges. We denote a subset of nodes $C^i \subseteq V$ as a cluster. A graph clustering algorithm then is a function $\text{CLUSTER} : (V, E) \rightarrow C$ such that $V = \bigcup_{C^i \in C} C^i$. We distinguish two classes of graph clustering algorithms: *hard* clustering algorithms (partitionings) produce non-overlapping clusters, that is, $C^i \cap C^j = \emptyset \iff i \neq j, \forall C^i, C^j \in C$, whereas *fuzzy* clustering algorithms permit cluster overlapping, that is, a node can be a member of several clusters in C .

⁴ <http://web.archive.org/web/20110327090414/http://labs.google.com/sets>.

⁵ A simple graph has no loops, i.e., $u \neq v, \forall \{u, v\} \in E$. We use this property for context disambiguation in Section 3.2.2.

**Figure 1**

The outline of the WATSET algorithm showing the *local* step of word sense induction and context disambiguation, and the *global* step of sense graph constructing and clustering.

3.1 Outline of WATSET, a Fuzzy Method for Local-Global Graph Clustering

WATSET constructs an intermediate representation of the input graph called a *sense graph*, which has been sketched as a “disambiguated word graph” in Biemann (2012). This is achieved by node sense induction based on hard clustering of the input graph node neighborhoods. The sense graph has the edges established between the different *senses* of the input graph nodes. The global clusters of the input graph are obtained by applying a hard clustering algorithm to the sense graph; removal of the sense labels yields overlapping clusters.

An outline of our algorithm is depicted in Figure 1. WATSET takes an undirected graph $G = (V, E)$ as the input and outputs a set of clusters C . The algorithm has two steps: local and global. The *local* step, as described in Section 3.2, disambiguates the potentially ambiguous nodes in G . The *global* step, as described in Section 3.3, uses these disambiguated nodes to construct an intermediate sense graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and produce the overlapping clustering C . WATSET is parameterized by two graph partitioning algorithms $\text{Cluster}_{\text{Local}}$ and $\text{Cluster}_{\text{Global}}$, and a context similarity measure sim . The complete pseudocode of WATSET is presented in Algorithm 1. For the sake of illustration, while describing the approach, we will provide examples with words and their synonyms. However, WATSET is not bound only to the lexical units and relationships, so our examples are given *without loss of generality*. Note also that WATSET can be applied for both unweighted and weighted graphs as soon as the underlying hard clustering algorithms $\text{Cluster}_{\text{Local}}$ and $\text{Cluster}_{\text{Global}}$ take edge weights into account.

3.2 Local Step: Node Sense Induction and Disambiguation

The *local* step of WATSET discovers the node senses in the input graph and uses this information to discover which particular senses of the nodes were connected via the edges of the input graph G .

3.2.1 Node Sense Induction. We induce node senses using the word neighborhood clustering approach by Dorow and Widdows (2003). In particular, we assume that the removal of the nodes participating in many triangles separates a graph into several

Algorithm 1 WATSET, a Local-Global Meta-Algorithm for Fuzzy Graph Clustering.

Input: graph $G = (V, E)$,
 hard clustering algorithms $\text{Cluster}_{\text{Local}}$ and $\text{Cluster}_{\text{Global}}$,
 context similarity measure $\text{sim} : (\text{ctx}(a), \text{ctx}(b)) \rightarrow \mathbb{R}, \forall \text{ctx}(a), \text{ctx}(b) \subseteq V$.

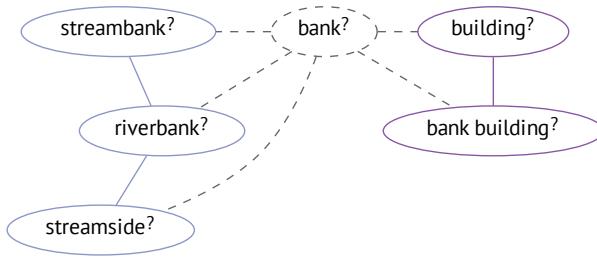
Output: clusters C .

```

1: for all  $u \in V$  do                                ▷ Local Step: Sense Induction
2:    $\text{senses}(u) \leftarrow \emptyset$ 
3:    $V_u \leftarrow \{v \in V : \{u, v\} \in E\}$            ▷ Note that  $u \notin V_u$ 
4:    $E_u \leftarrow \{\{v, w\} \in E : v, w \in V_u\}$ 
5:    $G_u \leftarrow (V_u, E_u)$ 
6:    $C_u \leftarrow \text{Cluster}_{\text{Local}}(G_u)$           ▷ Cluster the open neighborhood of  $u$ 
7:   for all  $C_u^i \in C_u$  do
8:      $\text{ctx}(u^i) \leftarrow C_u^i$ 
9:      $\text{senses}(u) \leftarrow \text{senses}(u) \cup \{u^i\}$ 
10:   $\mathcal{V} \leftarrow \bigcup_{u \in V} \text{senses}(u)$           ▷ Global Step: Sense Graph Nodes
11:  for all  $\hat{u} \in \mathcal{V}$  do                      ▷ Local Step: Context Disambiguation
12:     $\widehat{\text{ctx}}(\hat{u}) \leftarrow \emptyset$ 
13:    for all  $v \in \text{ctx}(\hat{u})$  do
14:       $\hat{v} \leftarrow \arg \max_{v' \in \text{senses}(v)} \text{sim}(\text{ctx}(\hat{u}) \cup \{u\}, \text{ctx}(v'))$       ▷  $\hat{u}$  is a sense of  $u \in V$ 
15:       $\widehat{\text{ctx}}(\hat{u}) \leftarrow \widehat{\text{ctx}}(\hat{u}) \cup \{\hat{v}\}$ 
16:   $\mathcal{E} \leftarrow \{\{\hat{u}, \hat{v}\} \in \mathcal{V}^2 : \hat{v} \in \widehat{\text{ctx}}(\hat{u})\}$           ▷ Global Step: Sense Graph Edges
17:   $\mathcal{G} \leftarrow (\mathcal{V}, \mathcal{E})$                       ▷ Global Step: Sense Graph Construction
18:   $C \leftarrow \text{Cluster}_{\text{Global}}(\mathcal{G})$           ▷ Global Step: Sense Graph Clustering
19:   $C \leftarrow \{\{u \in V : \hat{u} \in C^i\} \subseteq V : C^i \in C\}$           ▷ Remove the sense labels
20: return  $C$ 

```

connected components. Each component corresponds to the sense of the target node, so this procedure is executed for every node independently. Figure 2 illustrates this approach for sense induction. For related work on word sense induction approaches, see the survey in Section 2.2.

**Figure 2**

Clustering the neighborhood of the node “bank” of the input graph results in two clusters treated as the non-disambiguated sense contexts: $\text{bank}^1 = \{\text{streambank}, \text{riverbank}, \dots\}$ and $\{\text{bank}^2 = \text{bank building}, \text{building}, \dots\}$.

Table 2

Example of induced senses for the node “bank” and the corresponding clusters (contexts).

Sense	Context
$bank^1$	{streambank, riverbank, ...}
$bank^2$	{bank building, building, ...}
$bank^3$	{bank company, ...}
$bank^4$	{coin bank, penny bank, ...}

Given a node $u \in V$, we extract its open neighborhood $G_u = (V_u, E_u)$ from the input graph G , such that the target node u is not included into V_u (lines 3–5):

$$V_u = \{v \in V : \{u, v\} \in E\} \quad (1)$$

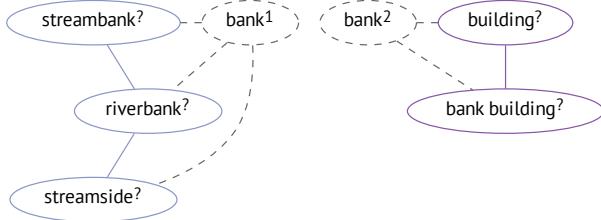
$$E_u = \{\{v, w\} \in E : v, w \in V_u\} \quad (2)$$

Then, we run a hard graph clustering algorithm on G_u that assigns one node to one and only one cluster, yielding a clustering C_u (line 6). We treat each obtained cluster $C_u^i \in C_u \subset V_u$ as representing a context for a different sense of the node $u \in V$ (lines 7–9). We denote, for example, $bank^1$, $bank^2$, and other labels as the node *senses* referred to as *senses(bank)*. In the example in Table 2, $|senses(bank)| = 4$. Given a sense $u^i \in senses(u)$, we denote $\text{ctx}(u^i) = C_u^i$ as a *context* of this sense of the node $u \in V$. Execution of this procedure for all the words in V results in the set of senses for the global step (line 10):

$$\mathcal{V} = \bigcup_{u \in V} \text{senses}(u) \quad (3)$$

3.2.2 Disambiguation of Neighbors. Although at the previous step we have induced node senses and mapped them to the corresponding contexts (Table 2), the elements of these contexts do not contain sense information. For example, the context of $bank^2$ in Figure 3 has two elements {bank building[?], building[?]}, the sense labels of which are currently not known. We recover the sense labels of nodes in a context using the sense disambiguated approach proposed by Faralli et al. (2016) as follows.

We represent each context as a vector in a vector space model (Salton, Wong, and Yang 1975) constructed for all the contexts. Because the graph G is simple (Section 3) and the context of any sense $\hat{u} \in \mathcal{V}$ does not include the corresponding node $u \in V$ (Table 2), we *temporarily* put it into context during disambiguation. This prevents the situation of non-matching when the context of a candidate sense $v' \in \text{senses}(v)$ has only one element and that element is u , that is, $\text{ctx}(v') = \{u\}$. We intentionally perform this insertion temporarily only during matching to prevent self-referencing. When a context $\text{ctx}(\hat{u}) \subset V$ is transformed into a vector, we assign to each element $v \in \text{ctx}(\hat{u})$ of this vector a weight equal to the weight of the edge $\{u, v\} \in E$ of the input graph G . If G is unweighted, we assign 1 if and only if $\{u, v\} \in E$, otherwise 0 is assigned. Table 3 shows an example of the context vectors used for disambiguating the word *building* in the context of the sense $bank^2$ in Figure 3. In this example the vectors essentially represent one-hot encoding as the example input graph is unweighted.

**Figure 3**

Contexts for two different senses of the node “bank”: only its senses bank^1 and bank^2 are currently known, whereas the other nodes in contexts need to be disambiguated.

Table 3

An example of context vectors for the node senses demonstrated in Figures 3 and 4. Because the graph is unweighted, one-hot encoding has been used. For matching purposes, the word “bank” is temporarily added into $\text{ctx}(\text{bank}^2)$.

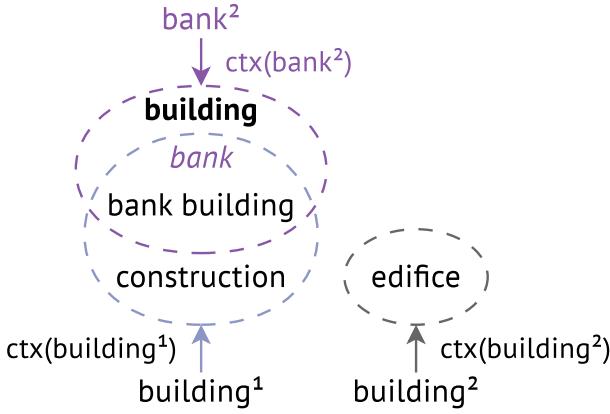
Sense	bank	bank building	building	construction	edifice
bank^2	1	1	1	0	0
building^1	1	1	0	1	0
building^2	0	0	0	0	1

Then, given a sense $\hat{u} \in \mathcal{V}$ of a node $u \in V$ and the context of this sense $\text{ctx}(\hat{u}) \subset V$, we *disambiguate* each node $v \in \text{ctx}(\hat{u})$. For that, we find the sense $\hat{v} \in \text{senses}(v)$ the context $\text{ctx}(\hat{v}) \subset V$, which maximizes the similarity to the target context $\text{ctx}(\hat{u})$. We compute the similarity using a context similarity measure $\text{sim} : (\text{ctx}(a), \text{ctx}(b)) \rightarrow \mathbb{R}$, $\forall \text{ctx}(a), \text{ctx}(b) \subseteq V$.⁶ Typical choices for the similarity measure are dot product, cosine similarity, Jaccard index, etc. Hence, we *disambiguate* each context element $v \in \text{ctx}(\hat{u})$:

$$\hat{v} = \arg \max_{v' \in \text{senses}(v)} \text{sim}(\text{ctx}(\hat{u}) \cup \{u\}, \text{ctx}(v')) \quad (4)$$

An example in Figure 4 illustrates the node sense disambiguation process. The context of the sense bank^2 is $\text{ctx}(\text{bank}^2) = \{\text{building}, \text{bank building}\}$ and the disambiguation target is *building*. Having chosen cosine similarity as the context similarity measure, we compute the similarity between $\text{ctx}(\text{bank}^2 \cup \{\text{bank}\})$ and the context of every sense of *building* in Table 3: $\cos(\text{ctx}(\text{bank}^2) \cup \{\text{bank}\}, \text{ctx}(\text{building}^1)) = \frac{2}{3}$ and $\cos(\text{ctx}(\text{bank}^2) \cup \{\text{bank}\}, \text{ctx}(\text{building}^2)) = 0$. Therefore, for the word *building* in the

⁶ For the sake of brevity, by *context similarity* we mean *similarity between context vectors in a sparse vector space model* (Salton, Wong, and Yang 1975).

**Figure 4**

Matching the meaning of the ambiguous node “building” in the context of the sense bank^2 . For matching purposes, the word “bank” is temporarily added into $\text{ctx}(\text{bank}^2)$.

context of bank^2 , its first sense, building^1 , should be used because its similarity value is higher.

Finally, we construct a disambiguated context $\widehat{\text{ctx}}(\hat{u}) \subset \mathcal{V}$ that is a sense-aware representation of $\text{ctx}(\hat{u})$. This disambiguated context indicates which node senses were connected to $\hat{u} \in \mathcal{V}$ in the input graph G . For that, in lines 13–15, we apply the disambiguation procedure defined in Equation (4) for every node $v \in \text{ctx}(\hat{u})$:

$$\widehat{\text{ctx}}(\hat{u}) = \{\hat{v} \in \mathcal{V} : v \in \text{ctx}(\hat{u})\} \quad (5)$$

As the result of the *local* step, for each node $u \in V$ in the input graph, we induce the $\text{senses}(u) \subset \mathcal{V}$ of nodes and provide each sense $\hat{u} \in \mathcal{V}$ with a disambiguated context $\widehat{\text{ctx}}(\hat{u}) \subseteq \mathcal{V}$.

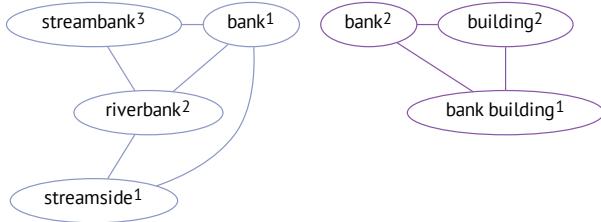
3.3 Global Step: Sense Graph Construction and Clustering

The *global* step of WATSET constructs an intermediate *sense graph* expressing the connections between the node senses discovered at the *local* step. We assume that the nodes \mathcal{V} of the sense graph are non-ambiguous, so running a hard clustering algorithm on this graph outputs clusters C covering the set of nodes V of the input graph G .

3.3.1 Sense Graph Construction. Using the set of node senses defined in Equation (3), we construct the sense graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ by establishing undirected edges between the senses connected through the disambiguated contexts (lines 16–17):

$$\mathcal{E} = \{\{\hat{u}, \hat{v}\} \in \mathcal{V}^2 : \hat{v} \in \widehat{\text{ctx}}(\hat{u})\} \quad (6)$$

Note that this edge construction approach disambiguates the edges E such that if a pair of nodes was connected in the input graph G , then the corresponding sense nodes will be connected in the sense graph \mathcal{G} . As a result, the constructed sense graph \mathcal{G} is a sense-aware representation of the input graph G . In the event G is weighted, we assign each edge $\{\hat{u}, \hat{v}\} \in \mathcal{E}$ the same weight as the edge $\{u, v\} \in E$ has in the input graph.

**Figure 5**

Clustering of the *sense graph* \mathcal{G} yields two clusters, $\{\text{bank}^1, \text{streambank}^3, \text{riverbank}^2, \dots\}$ and $\{\text{bank}^2, \text{bankbuilding}^1, \text{building}^2, \dots\}$; if one removes the sense labels, the clusters will overlap, resulting in a *soft* clustering of the input graph G .

3.3.2 Sense Graph Clustering. Running a hard clustering algorithm on \mathcal{G} produces the set of sense-aware clusters \mathcal{C} ; each sense-aware cluster $\mathcal{C}^i \in \mathcal{C}$ is a subset of \mathcal{V} (line 18). In order to obtain the set of clusters C that covers the set of nodes V of the input graph G , we simply remove the sense labels from the elements of clusters \mathcal{C} (line 19):

$$C = \{\{u \in V : \hat{u} \in \mathcal{C}^i\} \subseteq V : \mathcal{C}^i \in \mathcal{C}\} \quad (7)$$

Figure 5 illustrates the sense graph and its clustering in the example of the node “bank.” The construction of a sense graph requires disambiguation of the input graph nodes. Note that traditional approaches to graph-based sense induction, such as the ones proposed by Véronis (2004), Biemann (2006), and Hope and Keller (2013a), do not perform this step, but perform only local clustering of the graph because they do not aim at a global representation of clusters.

As the result of the *global* step, a set of clusters C of the input graph G is obtained, using an intermediate sense-aware graph \mathcal{G} . The presented local-global graph clustering approach, WATSET, makes it possible to naturally achieve a *soft* clustering of a graph using *hard* clustering algorithms only.

3.4 Simplified WATSET

The original WATSET algorithm, as previously published (Ustalov, Panchenko, and Biemann 2017) and described in Section 3.1, has context construction and disambiguation steps. These steps involve computation of a context similarity measure, which needs to be chosen as a hyper-parameter of the algorithm (Section 3.2.2). In this section, we propose a simplified version of WATSET (Algorithm 2) that requires no context similarity measure, which leads to faster computation in practice with less hyper-parameter tuning. As our experiments throughout this article show, this simplified version demonstrates similar performance to the original WATSET algorithm.

In the input graph G a pair of nodes $\{u, v\} \in V^2$ can be incident to one and only one edge. Otherwise, these nodes are not connected. Because of the use of a *hard* clustering algorithm for node sense induction (Section 2.2), in any pair of nodes $\{u, v\} \in E$, the node v can appear in the context of only one sense of u and vice versa. Therefore, we can omit the context disambiguation step (Section 3.2.2) by tracking the node sense identifiers produced during sense induction.

Algorithm 2 Simplified WATSET.

Input: graph $G = (V, E)$, hard clustering algorithms $\text{Cluster}_{\text{Local}}$ and $\text{Cluster}_{\text{Global}}$.**Output:** clusters C .

```

1:  $\mathcal{V} \leftarrow \emptyset$ 
2: for all  $u \in V$  do ▷ Local Step: Sense Induction
3:    $V_u \leftarrow \{v \in V : \{u, v\} \in E\}$  ▷ Note that  $u \notin V_u$ 
4:    $E_u \leftarrow \{\{v, w\} \in E : v, w \in V_u\}$ 
5:    $G_u \leftarrow (V_u, E_u)$ 
6:    $C_u \leftarrow \text{Cluster}_{\text{Local}}(G_u)$  ▷ Cluster the open neighborhood of  $u$ 
7:   for all  $C_u^i \in C_u$  do
8:     for all  $v \in C_u^i$  do ▷ Node  $v$  is connected to the  $i$ -th sense of  $u$ 
9:        $\text{senses}[u][v] \leftarrow i$ 
10:       $\mathcal{V} \leftarrow \mathcal{V} \cup \{u^i\}$ 
11:       $\mathcal{E} \leftarrow \{\{u^{\text{senses}[u][v]}, v^{\text{senses}[v][u]}\} \in \mathcal{V}^2 : \{u, v\} \in E\}$  ▷ Global Step: Sense Graph Edges
12:       $\mathcal{G} \leftarrow (\mathcal{V}, \mathcal{E})$  ▷ Global Step: Sense Graph Construction
13:       $C \leftarrow \text{Cluster}_{\text{Global}}(\mathcal{G})$  ▷ Global Step: Sense Graph Clustering
14:       $C \leftarrow \{\{u \in V : \hat{u} \in C^i\} \subseteq V : C^i \in \mathcal{C}\}$  ▷ Remove the sense labels
15: return  $C$ 

```

Given a pair $\{u, v\} \in E$, we reuse the sense information from Table 2 to determine which context of a sense $\hat{u} \in \mathcal{V}$ contains v . We denote this as $\text{senses}[u][v] \in \mathbb{N}$, which indicates $v \in \text{ctx}(u^{\text{senses}[u][v]})$, that is, the fact that node v is connected to the node u in the specified sense $u^{\text{senses}[u][v]}$. Following the example in Figure 2, if the context of *bank*¹ contains the word *streambank*, then the context of one of the senses of *streambank* must contain the word *bank* (e.g., *streambank*³). This information allows us to create Table 4, which allows producing the set of sense-aware edges by simultaneously retrieving the corresponding sense identifiers:

$$\mathcal{E} = \{\{u^{\text{senses}[u][v]}, v^{\text{senses}[v][u]}\} \in \mathcal{V}^2 : \{u, v\} \in E\} \quad (8)$$

This allows us to construct the sense graph \mathcal{G} in linear time $O(|E|)$ by querying the node sense index to disambiguate the input edges E in a deterministic way. Other steps are identical to the original WATSET algorithm (Section 3.1). Simplified WATSET is presented in Algorithm 2.

3.5 Algorithmic Complexity

We analyze the computational complexity of the separate routines of WATSET and then present the overall complexity compared with other hard and soft clustering algorithms. Our analysis is based on the assumption that the context similarity measure in Equation (4) can be computed in linear time with respect to the number of dimensions $d \in \mathbb{N}$. For instance, such measures as cosine and Jaccard satisfy this requirement. In all our experiments throughout this article, we use the cosine similarity measure: $\text{sim}(\text{ctx}(a), \text{ctx}(b)) = \cos(\text{ctx}(a), \text{ctx}(b))$, $\forall \text{ctx}(a), \text{ctx}(b) \subseteq V$. Provided that the context vectors are normalized, the complexity of such a measure is bound by the complexity of an inner product of two vectors, which is $O(|\text{ctx}(a) \cup \text{ctx}(b)|)$.

Table 4
Node sense identifier tracking in Simplified WATSET, according to Figure 2.

Source	Target	Index
bank	streambank	1
	riverbank	1
	streamside	1
	building	2
	bank building	2
streambank	bank	3
	riverbank	3
...		

Because the running time of our algorithm depends on the task-specific choice of two hard clustering algorithms, Cluster_{Local} and Cluster_{Global}, we report algorithm-specific analysis on two hard clustering algorithms that are popular in computational linguistics: CW by Biemann (2006) and MCL by van Dongen (2000). Given a graph $G = (V, E)$, the computational complexity is $O(|E|)$ for CW and $O(|V|^3)$ for MCL.⁷ Additionally, we denote \deg_{\max} as the maximum degree of G . Note that although, in general, \deg_{\max} is bound by $|V|$, in the real natural language-derived graphs this variable is distributed according to a power law. It is small for the majority of the nodes in a graph, making average running times acceptable in practice, as presented in Section 3.5.5.

3.5.1 Node Sense Induction. This operation is executed for every node of the input graph G , that is, $|V|$ times. By definition of an undirected graph, the maximum number of neighbors of a node in G is \deg_{\max} and the maximum number of edges in a neighborhood is $\frac{\deg_{\max}(\deg_{\max}-1)}{2}$. Thus, this operation takes $O(|V|\deg_{\max}^2)$ steps with CW and $O(|V|\deg_{\max}^3)$ steps with MCL.

3.5.2 Disambiguation of Neighbors. Let $senses_{\max}$ be the maximum number of senses for a node and ctx_{\max} be the maximum size of the node sense context. Thus, this operation takes $O(|V| \times senses_{\max} \times ctx_{\max})$ steps to iterate over all the node sense contexts. At each iteration, it scans all the senses of the ambiguous node in context and computes a similarity between its context and the candidate sense context in a linear time (Section 3.5). This requires $O(senses_{\max} \times ctx_{\max})$ steps per each node in context. Therefore, the whole operation takes $O(|V| \times senses_{\max}^2 \times ctx_{\max}^2)$ steps. Because the maximum number of node senses is observed in a special case when the neighborhood is an unconnected graph, $senses_{\max} \leq \deg_{\max}$. Given the fact that the maximum context size is observed in a special case when the neighborhood is a fully connected graph, $ctx_{\max} \leq$

⁷ Although MCL can be implemented more efficiently than $O(|V|^3)$, cf. van Dongen (2000, page 125), we would like to use the consistent worst case scenario notation for all the mentioned clustering algorithms.

Table 5

Computational complexity of graph clustering algorithms, where $|V|$ is the number of vertices, $|E|$ is the number of edges, and \deg_{\max} is the maximum degree of a vertex. For brevity, we do not insert rows corresponding to Simplified WATSET (Algorithm 2), which does not require the $O(|V| \deg_{\max}^4)$ term related to context disambiguation.

Algorithm	Hard or Soft	Computational Complexity
Chinese Whispers (Biemann 2006)	hard	$O(E)$
Markov Clustering (van Dongen 2000)	hard	$O(V ^3)$
MaxMax (Hope and Keller 2013a)	soft	$O(E)$
Louvain method (Blondel et al. 2008)	hard	$O(V \log(V))$
Clique Percolation (Palla et al. 2005)	soft	$2^{ V }$
WATSET[CW, CW]	soft	$O(V ^2 \deg_{\max}^2 + V \deg_{\max}^4)$
WATSET[CW, MCL]	soft	$O(V ^3 \deg_{\max}^3 + V \deg_{\max}^4)$
WATSET[MCL, CW]	soft	$O(V ^2 \deg_{\max}^2 + V \deg_{\max}^4)$
WATSET[MCL, MCL]	soft	$O(V ^3 \deg_{\max}^3 + V \deg_{\max}^4)$

\deg_{\max} . Thus, disambiguation of all the node sense contexts takes $O(|V| \deg_{\max}^4)$ steps. Note that because the simplified version of WATSET, as described in Section 3.4, does not perform context disambiguation, this term should be taken into account only for the original version of WATSET (Algorithm 1).

3.5.3 Sense Graph Clustering. Like the input graph G , the sense graph \mathcal{G} is undirected, so it has at most $|V| \deg_{\max}$ nodes and $\frac{|V| \deg_{\max}(|V| \deg_{\max} - 1)}{2}$ edges. Thus, this operation takes $O(|V|^2 \deg_{\max}^2)$ steps with CW and $O(|V|^3 \deg_{\max}^3)$ steps with MCL.

3.5.4 Overall Complexity. Table 5 presents a comparison of WATSET to other hard and soft graph clustering algorithms popular in computational linguistics,⁸ such as CW by Biemann (2006), MCL by van Dongen (2000), and MaxMax by Hope and Keller (2013a). Additionally, we compare WATSET with several graph clustering algorithms that are popular in network science, such as the Louvain method by Blondel et al. (2008) and CPM by Palla et al. (2005). The notation WATSET[MCL, CW] means using MCL for local clustering and CW for global clustering (cf. the discussion on graph clustering algorithms in Section 2.1).

The analysis shows that the most time-consuming operations in WATSET are sense graph clustering and context disambiguation. Although the overall computational complexity of our meta-algorithm is higher than that of the other methods, its compute-intensive operations, such as node sense induction and context disambiguation, are

⁸ Our survey was based on Mihalcea and Radev (2011), Di Marco and Navigli (2013), and Lewis and Steedman (2013a).

Table 6

Parameters of the co-occurrence graphs for different corpus sizes in the Leipzig Corpora Collection, where $|V|$ is the number of vertices, $|E|$ is the number of edges, and \deg_{\max} is the maximum degree of a vertex; time is measured in minutes.

Size	$ V $	$ E $	\deg_{\max}	Sequential Time, min.	Parallel Time, min.
10K	4,907	16,057	547	0.13 ± 0.01	0.04 ± 0.00
30K	11,627	55,181	1,307	0.91 ± 0.05	0.36 ± 0.02
100K	27,200	203,946	3,319	9.33 ± 0.13	3.78 ± 0.08
300K	55,359	630,138	7,467	53.34 ± 0.16	24.44 ± 0.18
1M	117,141	2,031,283	18,081	347.16 ± 1.97	158.00 ± 1.88

executed for every node independently, so the algorithm can easily be run in a parallel or a distributed way to reduce the running time.

3.5.5 An Empirical Evaluation of Average Running Times. In order to evaluate the running time of WATSET on a real-world scenario, we applied it to the clustering of co-occurrence graphs. Word clusters discovered from co-occurrence graphs are the sets of semantically related polysemous words, so we ran our sense-aware clustering algorithm to obtain overlapping word clusters.

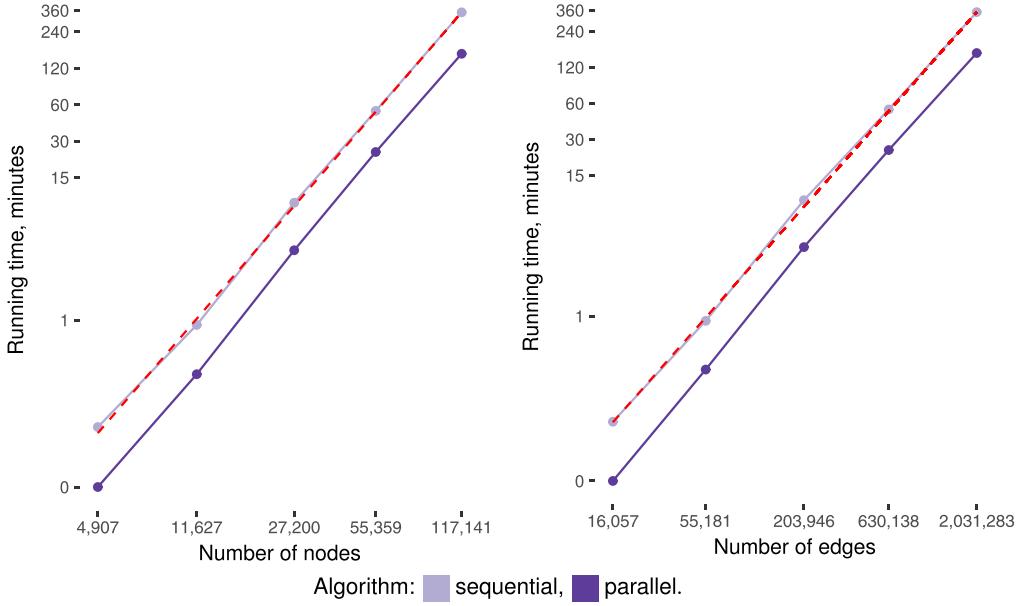
We used the English word co-occurrence graphs from the Leipzig Corpora Collection by Goldhahn, Eckart, and Quasthoff (2012) because it is partitioned into corpora of different sizes.⁹ We evaluated the graphs corresponding to five different English corpus sizes: 10K, 30K, 100K, 300K, and 1M tokens (Table 6). The measurements were made independently among the graphs using the WATSET[CW, CW] algorithm with the lowest complexity bound by $O(|V|^2 \deg_{\max}^2 + |V| \deg_{\max}^4)$.

Because our implementation of WATSET in the Java programming language, as described in Section 7, is multi-threaded and runs node sense induction and context disambiguation steps in parallel, we study the benefit of multiple available central processing unit (CPU) cores to the overall running time. The single-threaded setup that uses only one CPU core will be referred to as *sequential*, while the multi-threaded setup that uses all the CPU cores available on the machine will be referred to as *parallel*.

For each graph, we ran WATSET five times. Following Horký et al. (2015), the first three runs were used off-record to *warm-up* the Java virtual machine. The next two runs were used for actual measurement. We used the following computational node for this experiment: two Intel Xeon E5-2630 v4 CPUs, 256 GB of ECC RAM, Ubuntu 16.04.4 LTS (Linux 4.13.0, x86_64), Oracle Java 8b121; 40 logical cores were available in total. Table 6 reports the running time mean and the standard deviation for both setups, sequential and parallel.

Figure 6 shows the polynomial growth of $O(|V|^{2.52})$, which is smaller than the worst case of $O(|V|^2 \deg_{\max}^2 + |V| \deg_{\max}^4)$. This is because in co-occurrence graphs, as well as in many other real-world graphs that also exhibit scale-free small world properties (Steyvers and Tenenbaum 2005), the degree distribution among nodes is strongly right-skewed. This makes WATSET useful for processing real-world graphs. Both Table 6

⁹ <http://wortschatz.uni-leipzig.de/en/download>.

**Figure 6**

Log-log plots showing growth of the empirical average running time in number of nodes (left) and number of edges (right) of two WATSET[CW_{top}, CW_{top}] setups: **sequential** and **parallel**. The **dashed** line is fitted to the running time data of the sequential version of WATSET, showing polynomial growth in $O(|V|^{2.52})$ and $O(|E|^{1.63})$, respectively.

and Figure 6 clearly confirm that WATSET scales well and can be parallelized on multiple CPU cores, which makes it possible to process very large graphs.

4. Application to Unsupervised Synset Induction

A *synset* is a set of mutual synonyms, which can be represented as a clique graph where nodes are words and edges are synonymy relationships. Synsets represent word senses and are building blocks of thesauri and lexical ontologies, such as WordNet (Fellbaum 1998). These resources are crucial for many NLP applications that require common sense reasoning, such as information retrieval (Gong, Cheang, and Hou U 2005), sentiment analysis (Montejo-Ráez et al. 2014), and question answering (Kwok, Etzioni, and Weld 2001; Zhou et al. 2013).

For most languages, no manually constructed resource is available that is comparable to the English WordNet in terms of coverage and quality (Braslavski et al. 2016). For instance, Kiselev, Porshnev, and Mukhin (2015) present a comparative analysis of lexical resources available for the Russian language, concluding that there is no resource compared with WordNet in terms of completeness and availability for Russian. This lack of linguistic resources for many languages strongly motivates the development of new methods for automatic construction of WordNet-like resources. In this section, we apply WATSET for unsupervised synset induction from a synonymy graph and compare it with state-of-the-art graph clustering algorithms run on the same task.

4.1 Synonymy Graph Construction and Clustering

Wikipedia,¹⁰ Wiktionary,¹¹ OmegaWiki,¹² and other collaboratively created resources contain a large amount of lexical semantic information—yet are designed to be human-readable and not formally structured. Although semantic relationships can be automatically extracted using tools such as DKPro JWKT¹³ by Zesch, Müller, and Gurevych (2008) and Wikokit¹⁴ by Krizhanovsky and Smirnov (2013), words in these relationships are not disambiguated. For instance, the synonymy pairs *{bank, streambank}* and *{bank, banking company}* will be connected via the word “bank,” although they refer to different senses. This problem stems from the fact that articles in Wiktionary and similar resources list “undisambiguated” synonyms. They are easy to disambiguate for humans while reading a dictionary article but can be a source of errors for language processing systems.

Although large-scale automatically constructed lexical semantic resources like BabelNet (Navigli and Ponzetto 2012) are available, they contain synsets with relationships other than synonymity. For instance, in BabelNet 4.0, the synset for *bank* as an institution contains among other things non-synonyms like *Monetary intermediation* and *Money-lenders*.¹⁵

A synonymy dictionary can be perceived as a graph, where the nodes correspond to lexical units (words) and the edges connect pairs of the nodes when the synonymy relationship between them holds. Because such a graph can easily be obtained for arbitrary language, we expect that constructing and clustering a sense-aware representation of a synonymy graph yields plausible synsets covering polysemous words.

4.1.1 Synonymy Graph Construction. Given a synonymy dictionary, we construct the synonymy graph $G = (V, E)$ as follows. The set of nodes V includes every lexical unit appearing in the input dictionary. An edge in the set of edges $E \subseteq V^2$ is established if and only if a pair of words are distinguished synonyms, according to the input synonymy dictionary. To enhance our representation with the contextual semantic similarity between synonyms, we assigned every edge $\{u, v\} \in E$ a weight equal to the cosine similarity of Skip-Gram word embeddings (Mikolov et al. 2013). As a result, we obtained a weighted synonymy graph G .

4.1.2 Synonymy Graph Clustering. Because the graph G contains both monosemeous and polysemous words without indication of the particular senses, we run WATSET to obtain a soft clustering C of the synonymy graph G . Since our algorithm explicitly induces and clusters the word senses, the elements of the clusters C are by definition synsets, that is, sets of words that are synonymous with each other.

4.2 Evaluation

We conduct our experiments on resources from two different languages. We evaluate our approach on two data sets for English to demonstrate its performance in a

10 <http://www.wikipedia.org>.

11 <http://www.wiktionary.org>.

12 <http://www.omegawiki.org>.

13 <https://dkpro.github.io/dkpro-jwktl>.

14 <https://github.com/componavt/wikokit>.

15 <https://babelnet.org/synset?word=bn:00008364n>.

resource-rich language. Additionally, we evaluate it on two Russian data sets, because Russian is a good example of an under-resourced language with a clear need for synset induction (Kiselev, Porshnev, and Mukhin 2015).

4.2.1 Experimental Set-Up. We compare WATSET with five popular graph clustering methods presented in Section 2.1: CW, MCL, MaxMax, ECO, and the CPM. The first two algorithms perform *hard* clustering algorithms, and the last three are *soft* clustering methods just like our method. Although the hard clustering algorithms are able to discover clusters that correspond to synsets composed of unambiguous words, they can produce wrong results in the presence of lexical ambiguity when a node should belong to several synsets. In our experiments, we use CW and MCL also as the underlying algorithms for local and global clustering in WATSET, so our comparison will show the difference between the “plain” underlying algorithms and their utilization in WATSET. We also report the performance of Simplified WATSET (Section 3.4).

In our experiments, we rely on our own implementation of MaxMax and ECO, as reference implementations are not available. For CW,¹⁶ MCL,¹⁷ and CPM,¹⁸ available implementations have been used. During the evaluation, we delete clusters equal to or larger than the threshold of 150 words, as they can hardly represent any meaningful synset. Only the clusters produced by the MaxMax algorithm were actually affected by this threshold.

Quality Measure. To evaluate the quality of the induced synsets, we transform them into synonymy pairs and computed precision, recall, and F_1 -score on the basis of the overlap of these synonymy pairs with the synonymy pairs from the gold standard data sets. The F_1 -score calculated this way is known as **paired F-score** (Manandhar et al. 2010; Hope and Keller 2013a). Let C be the set of obtained synsets and C_G be the set of gold synsets. Given a synset containing $n > 1$ words, we generate $\frac{n(n-1)}{2}$ pairs of synonyms, so we transform C into a set of pairs P and C_G into a set of gold pairs P_G . We then compute the numbers of positive and negative answers as follows:

$$TP = |P \cup P_G| \quad (9)$$

$$FP = |P \setminus P_G| \quad (10)$$

$$FN = |P_G \setminus P| \quad (11)$$

where TP is the number of true positives, FP is the number of false positives, and FN is the number of false negatives. As a result, we use the standard definitions of precision as $Pr = \frac{TP}{TP+FP}$, recall as $Re = \frac{TP}{TP+FN}$, and F_1 -score as $F_1 = \frac{2 \cdot Pr \cdot Re}{Pr + Re}$. The advantage of this measure compared with other cluster evaluation measures, such as *fuzzy B-Cubed* (Jurgens and Klapaftis 2013) and *normalized modified purity* (Kawahara, Peterson, and Palmer 2014), is its straightforward interpretability.

Statistical Testing. We evaluate the statistical significance of the experimental results using a McNemar’s test (1947). Given the results of two algorithms, we build a 2×2 contingency table and compute the p-value of the test using the Statsmodels

¹⁶ <https://github.com/uhh-lt/chinese-whispers>.

¹⁷ <https://micans.org/mcl/>.

¹⁸ <https://networkx.github.io>.

Table 7

Statistics of the gold standard data sets used in our experiments.

Resource	Language	# words	# synsets	# pairs
WordNet	English	148,730	117,659	152,254
BabelNet		11,710,137	6,667,855	28,822,400
RuWordNet	Russian	110,242	49,492	278,381
YARN		9,141	2,210	48,291

toolkit (Seabold and Perktold 2010).¹⁹ Since the hypothesis tested by the McNemar’s test is whether the results from both algorithms are similar against the alternative that they are not, we use the p-value of this test to assess the significance of the difference between F_1 -scores (Dror et al. 2018). We consider the performance of one algorithm to be higher than the performance of another if its F_1 -score is larger and the corresponding p-value is smaller than a significance level of 0.01.

Gold Standards. We conduct our evaluation on four lexical semantic resources for two different natural languages. Statistics of the gold standard data sets are present in Table 7. We report the number of lexical units (# words), synsets (# synsets), and the generated synonymy pairs (# pairs).

We use WordNet,²⁰ a popular *English* lexical database constructed by expert lexicographers (Fellbaum 1998). WordNet contains general vocabulary and appears to be the de facto gold standard in similar tasks (Hope and Keller 2013a). We used WordNet 3.1 to derive the synonymy pairs from synsets. Additionally, to compare to an automatically constructed lexical resource, we use BabelNet,²¹ a large-scale multilingual semantic network based on WordNet, Wikipedia, and other resources (Navigli and Ponzetto 2012). We retrieved all the synonymy pairs from the BabelNet 3.7 synsets marked as English, using the BabelNet Extract tool (Ustalov and Panchenko 2017).

As a lexical ontology for *Russian*, we use RuWordNet²² by Loukachevitch et al. (2016), containing both general vocabulary and domain-specific synsets related to sport, finance, economics, and so forth. Up to one half of the words in this resource are multi-word expressions (Kiselev, Porshnev, and Mukhin 2015), which is due to the coverage of domain-specific vocabulary. RuWordNet is a WordNet-like version of the RuThes thesaurus that is constructed in the traditional way, namely by a small group of expert lexicographers (Loukachevitch 2011). In addition, we use Yet Another RussNet²³ (YARN) by Braslavski et al. (2016) as another gold standard for Russian. The resource is constructed using crowdsourcing and mostly covers general vocabulary. In particular, non-expert users are allowed to edit synsets in a collaborative way, loosely supervised by a team of project curators. Because of the ongoing development of the resource, we selected as the silver standard only those synsets that were edited at least eight times

19 <https://www.statsmodels.org/>.

20 <https://wordnet.princeton.edu>.

21 <https://www.babelnet.org>.

22 <https://ruwordnet.ru/en>.

23 <https://russianword.net/en>.

Table 8

Statistics of the input data sets used in our experiments.

Language	# words	# pairs
English	243,840	212,163
Russian	83,092	211,986

in order to filter out noisy incomplete synsets.²⁴ We do not use BabelNet for evaluating the Russian synsets, as our manual inspection during prototyping showed, on average, a much lower quality than its English subset.

Input Data. For each language, we constructed a synonymy graph using openly available synonymy dictionaries. The statistics of the graphs used as the input in the further experiments are shown in Table 8.

For *English*, synonyms were extracted from the English Wiktionary,²⁵ which is the largest Wiktionary at the present moment in terms of the lexical coverage, using the DKPro JWKT tool by Zesch, Müller, and Gurevych (2008). English words have been extracted from the dump.

For *Russian*, synonyms from three sources were combined to improve lexical coverage of the input dictionary and to enforce confidence in jointly observed synonyms: (1) synonyms listed in the Russian Wiktionary extracted using the Wikokit tool by Krizhanovsky and Smirnov (2013); (2) the dictionary of Abramov (1999); and (3) the Universal Dictionary of Concepts (Dikonov 2013). Whereas the two latter resources are specific to Russian, Wiktionary is available for most languages. Note that the same input synonymy dictionaries were used by authors of YARN to construct synsets using crowdsourcing. The results on the YARN data set show how closely an automatic synset induction method can approximate manually created synsets provided the same starting material.²⁶

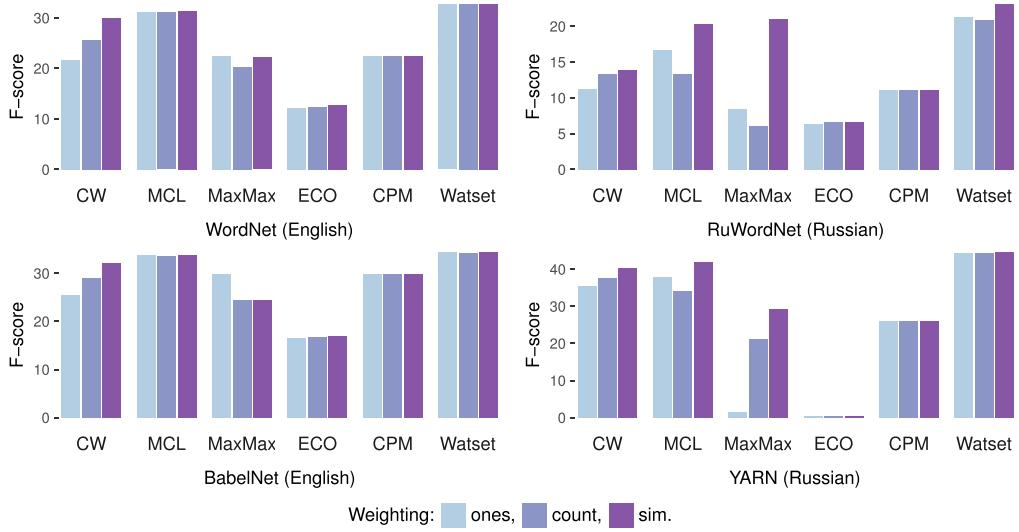
Because of the vocabulary differences between the input data and the gold standard data sets, we use the intersection between the lexicon of the gold standard and the united lexicon of all the compared configurations of the algorithms during all the experiments in this section.

4.2.2 Parameter Tuning. We tuned the hyper-parameters for such methods as CPM (Palla et al. 2005) and ECO (Gonçalo Oliveira and Gomes 2014) on the evaluation data set. We do not perform any tuning of WATSET because the underlying local and global clustering algorithms, CW and MCL, are parameter-free, so we use default configurations of these and their variations. As $CPM_{k=3}$ we denote that this method showed the best performance using the threshold value of $k = 3$. For ECO, we found the threshold value of $\theta = 0.05$ yielding the best results, as opposed to the value of $\theta = 0.2$ suggested by Gonçalo Oliveira and Gomes (2014).

²⁴ In YARN, an edit operation can be an addition or a removal of a synset element; an average synset in our data set contains 6.77 ± 3.54 words.

²⁵ We used the Wiktionary dumps of February 1, 2017.

²⁶ We used the YARN dumps of February 7, 2017.

**Figure 7**

Impact of the different graph-weighting schemas on the performance of synset induction. Each bar corresponds to the top performance of a method in Tables 9 and 10.

We also study the performance impact of different edge-weighting approaches for the same input graph. For that, we present the results of running the same algorithms in three different setups: ones that assigns every edge the constant weight of 1, count that weights the edge $\{u, v\} \in E$ with the number of times a synonymy pair appeared in the input dictionary, and sim that uses cosine similarity between word embeddings, as described in Section 4.1.1. For English, we use the commonly used 300-dimensional word embeddings trained on the 100 billion tokens Google News corpus.²⁷ For Russian, we use the 500-dimensional embeddings from the Russian Distributional Thesaurus trained on a 12.9 billion tokens corpus of books, which yielded the state-of-art performance on a shared task on Russian semantic similarity (Panchenko et al. 2017).²⁸

4.2.3 Results and Discussion. Figure 7 presents an overview of the evaluation results on both data sets. Because the synonymy graph construction step is the same for all the experiments, we start our analysis with the comparison of different edge-weighting approaches introduced in Section 4.2.2: constant values (ones), frequencies (count), and semantic similarity scores (sim) based on word vector similarity. Results across various configurations and methods indicate that using the weights based on the similarity scores provided by word embeddings is the best strategy for all methods except MaxMax on the English data sets. However, its performance using the ones weighting does not exceed the other methods using the sim weighting. Therefore, we report all further results on the basis of the sim weights. The edge weighting scheme impacts Russian more for most algorithms. The CW algorithm, however, remains sensitive to the weighting also for the English data set due to its randomized nature.

27 <https://code.google.com/archive/p/word2vec/>.

28 <https://doi.org/10.5281/zenodo.163857>.

Table 9

Comparison of the synset induction methods on data sets for English. All methods rely on the similarity edge weighting (sim); best configurations of each method in terms of F_1 -scores are shown for each data set. Results are sorted by F_1 -score on BabelNet; top three values of each measure are **boldfaced**, and statistically significant results are marked with an asterisk (*). Simplified WATSET is denoted as WATSET\$.

Method	# words	# synsets	# pairs	WordNet			BabelNet		
				Pr	Re	F_1	Pr	Re	F_1
WATSET[MCL, MCL]	243,840	112,267	345,883	34.48	30.82	32.54*	40.01	30.06	34.33*
MCL	243,840	84,679	387,315	34.21	29.10	31.45*	38.98	29.97	33.89*
CW _{top}	243,840	77,879	539,753	28.54	31.67	30.02*	32.57	31.71	32.14*
WATSET[CW _{log} , MCL]	243,840	164,689	227,906	39.35	27.99	32.71*	43.94	24.47	31.44*
WATSET\$(CW _{top} , MCL)	243,840	164,683	227,872	39.17	27.83	32.54*	43.87	24.40	31.36*
WATSET\$(CW _{log} , MCL)	243,840	165,406	222,554	40.20	27.44	32.62*	44.63	24.09	31.29*
CPM _{k=2}	186,896	67,109	317,293	56.06	14.06	22.48*	49.23	21.44	29.87*
MaxMax	219,892	73,929	797,743	17.59	29.97	22.17*	20.16	31.34	24.53*
ECO	243,840	171,773	84,372	78.41	6.95	12.77	69.91	9.59	16.87

Tables 9 and 10 present evaluation results for both languages. For each method, we show the best configurations in terms of F_1 -score. One may note that the granularity of the resulting synsets, especially for Russian, is very different, ranging from 4,000 synsets for the CPM_{k=3} method to 67,645 induced by the ECO method. Both tables report the number of words, synsets, and synonyms after pruning huge clusters larger than 150 words. Without this pruning, the MaxMax and CPM methods tend to discover giant components obtaining almost zero precision as we generate all possible pairs of nodes in such clusters. The other methods did not exhibit such behavior.

The disambiguation of the input graph performed by the WATSET method splits nodes belonging to several local communities to several nodes, significantly facilitating the clustering task otherwise complicated by the presence of the hubs that wrongly link semantically unrelated nodes. WATSET robustly outperformed all other methods,

Table 10

Results on data sets for Russian sorted by F_1 -score on Yet Another RussNet (YARN); top three values of each measure are **boldfaced** and statistically significant results are marked with an asterisk (*). Simplified WATSET is denoted as WATSET\$.

Method	# words	# synsets	# pairs	RuWordNet			YARN		
				Pr	Re	F_1	Pr	Re	F_1
WATSET\$(CW _{lin} , MCL]	83,092	58,353	242,615	15.01	32.55	20.55*	46.70	42.69	44.61*
WATSET[CW _{lin} , MCL]	83,092	55,369	332,727	11.95	34.91	17.81*	40.10	46.32	42.99*
MCL	83,092	21,973	353,848	15.54	29.10	20.26*	54.95	33.94	41.97*
CW _{lin}	83,092	19,124	672,076	8.73	34.20	13.91*	36.33	45.13	40.25*
WATSET\$(MCL, CW _{lin})	83,092	62,700	175,643	19.46	28.48	23.12*	52.28	29.41	37.65*
MaxMax	83,092	27,011	461,748	17.58	26.09	21.01*	58.24	19.49	29.20*
CPM _{k=3}	15,555	4,000	45,231	23.44	7.23	11.05*	62.51	6.04	11.02*
ECO	83,092	67,645	18,362	72.41	3.45	6.58	90.36	0.18	0.36

according to the F_1 -score on all the data sets for English (Table 9) and Russian (Table 10). In particular, on WordNet for English, WATSET[CW_{log}, MCL] has statistically significantly outperformed all other methods ($p \ll 0.01$), including different configurations of our algorithm. On BabelNet for English, WATSET[MCL, MCL] showed a similar behavior ($p \ll 0.01$). On RuWordNet for Russian, Simplified WATSET[MCL, CW_{lin}] statistically significantly outperformed all other algorithms, including highly competitive MCL and MaxMax ($p \ll 0.01$). Similarly, on YARN for Russian, Simplified WATSET[CW_{lin}, MCL] has significantly outperformed all the other algorithms ($p \ll 0.01$).

Interestingly, in all the cases, the toughest competitor was a hard clustering algorithm—MCL (van Dongen 2000). We observed that the “plain” MCL successfully groups monosemous words, but isolates the neighborhood of polysemous words, which results in the recall drop in comparison to WATSET. CW operates more quickly due to a simplified update step. On the same graph, CW tends to produce larger clusters than MCL. This leads to a higher recall of “plain” CW as compared with the “plain” MCL, at the cost of lower precision. Although MCL demonstrated highly competitive results, the best configuration of WATSET has statistically significantly outperformed it on all the data sets.

Using MCL instead of CW for sense induction in WATSET expectedly produced more fine-grained senses. However, at the global clustering step, these senses erroneously tend to form coarse-grained synsets connecting unrelated senses of the ambiguous words. This explains the generally higher recall of WATSET[MCL, ·]. Despite the randomized nature of CW, variance across runs do not affect the overall ranking. The rank of different weighting schemes on the node degree of CW_{top/lin/log} can change, while the rank of the best CW configuration compared to other methods remains the same.

The MaxMax algorithm showed mixed results. On the one hand, it outputs large clusters uniting more than a hundred nodes. This inevitably leads to a high recall, as it is clearly seen in the results for Russian because such synsets still pass under our cluster size threshold of 150 words. Its synsets on the English data sets are even larger and have been pruned, which resulted in the low recall. On the other hand, smaller synsets having at most 10–15 words were identified correctly. MaxMax appears to be extremely sensitive to edge weighting, which also complicates its application in practice.

The CPM algorithm showed unsatisfactory results, emitting giant components encompassing thousands of words. Such clusters were automatically pruned, but the remaining clusters are quite correct synsets, which is confirmed by the high precision values. When increasing the minimal number of elements in the clique k , recall improves, but at the cost of a dramatic precision drop. We suppose that the network structure assumptions exploited by CPM do not accurately model the structure of our synonymy graphs.

Finally, the ECO method yielded the worst results because most of the cluster candidates failed to pass through the constant threshold used for estimating whether a pair of words should be included in the same cluster. Most synsets produced by this method were trivial (i.e., containing only a single word). The remaining synsets for both languages have at most three words that have been connected by chance due to the edge noising procedure used in this method, resulting in a low recall.

The results obtained on all gold standards (Figure 7) show similar trends in terms of relative ranking of the methods. Yet absolute scores of YARN and RuWordNet are substantially different because of the inherent difference of these data sets. RuWordNet is more domain-specific in terms of vocabulary, so our input set of generic synonymy dictionaries has a limited coverage on this data set. On the other hand, recall calculated

Table 11

Sample synsets induced by the WATSET[MCL, MCL] method for English using the sim weighting approach.

Size	Synset
2	decimal point, dot
2	wall socket, power point
3	gullet, throat, food pipe
3	CAT, computed axial tomography, CT
4	microwave meal, ready meal, TV dinner, frozen dinner
4	mock strawberry, false strawberry, gurbir, Indian strawberry
5	objective case, accusative case, oblique case, object case, accusative
5	discipline, sphere, area, domain, sector
6	radio theater, dramatized audiobook, audio theater, radio play, radio drama, audio play
6	integrator, reconciler, consolidator, mediator, harmonizer, uniter
7	invite, motivate, entreat, ask for, incentivize, ask out, encourage
7	curtail, craw, yield, riding crop, harvest, crop, hunting crop

on YARN is substantially higher as this resource was manually built on the basis of synonymy dictionaries used in our experiments.

Table 11 presents examples of the obtained synsets of various sizes for the top WATSET configuration on English. As one might observe, the quality of the results is highly plausible. Because in this configuration we assigned edge weights based on the cosine of the angle between Skip-Gram word vectors (Mikolov et al. 2013), we should note that such an approach assigns high values of similarity not just to synonymous words, but to antonymous and generally any lexically related words. This is a common problem with lexical embedding spaces, which we tried to evade by explicitly using a synonymy dictionary as an input. For example, “audio play” and “radio play,” or “accusative” and “oblique,” are semantically related expressions, but really not synonyms. Such a problem can be addressed using techniques such as retrofitting (Faruqui et al. 2015) and contextualization (Peters et al. 2018).

However, one limitation of all the approaches considered in this section is the dependence on the completeness of the input dictionary of synonyms. In some parts of the input synonymy graph, important bridges between words can be missing, leading to smaller-than-desired synsets. A promising extension of the present methodology is using distributional models to enhance connectivity of the graph by cautiously adding extra relationships (Ustalov et al. 2017).

Cross-Resource Evaluation. In order to estimate the upper bound of precision, recall, and F_1 -score in our synset induction experiments, we conducted a cross-resource evaluation between the used gold-standard data sets (Table 12). Similarly to the experimental setup described in Section 4.2.1, we transformed synsets from every data set into sets of synonymy pairs. Then, for every pair of gold standard data sets, we computed the pairwise precision, recall, and F_1 -score by assessing synset-induced synonymy pairs of one data set on the pairs of another data set. As a result, we see that the low absolute numbers in evaluation are due to an inherent vocabulary mismatch between the input dictionaries of synonyms and the gold data sets because no single resource for Russian can obtain high recall scores on another one. Surprisingly, even BabelNet, which integrates most

Table 12

Performance of lexical resources cross-evaluated against each other.

Input Synsets	Gold Synsets	Language	Pr	Re	F ₁
BabelNet	WordNet	English	72.93	99.76	84.26
WordNet	BabelNet		99.79	69.86	82.18
YARN	RuWordNet	Russian	16.36	16.21	16.28
BabelNet	RuWordNet		34.84	40.87	37.61
RuWordNet	YARN	Russian	66.96	12.13	20.54
BabelNet	YARN		51.53	10.89	17.98

of the available lexical resources, still does not reach a recall substantially larger than 50%.²⁹ Note that the results of this cross-data set evaluation are not directly comparable to results in Table 10 since in our experiments we use much smaller input dictionaries than those used by BabelNet. Our cross-resource evaluation demonstrates that unlike WordNet and BabelNet, which are built on a similar conceptual basis, RuWordNet and YARN have a very different structure, so an algorithm that shows good results on one will likely not perform very well on another.

5. Application to Unsupervised Semantic Frame Induction

In this section, our goal is to investigate the applicability of our graph clustering technique in a different task. Namely, we explore how *semantic frames*—more complex linguistic structures than synsets—can be induced from text using WATSET. A **semantic frame** is a central concept of the Frame Semantics theory (Fillmore 1982). A frame is a structure that describes a certain situation or action (e.g., “Dining” or “Kidnapping”) in terms of participants involved in these actions, which fill semantic roles of this frame and words commonly describing such situations. Figure 8 illustrates a part of the “Kidnapping” semantic frame from the FrameNet resource.³⁰

Recent years have seen much work on frame semantics, enabled by the availability of a large set of frame definitions, as well as a manually annotated text corpus provided by the FrameNet project (Baker, Fillmore, and Lowe 1998). FrameNet data enabled the development of wide-coverage frame parsers using supervised learning (Gildea and Jurafsky 2002; Erk and Padó 2006; Das et al. 2014, *inter alia*), as well as its application to a wide range of tasks, ranging from answer extraction in Question Answering (Shen and Lapata 2007) and Textual Entailment (Burchardt et al. 2009; Ben Aharon, Szpektor, and Dagan 2010), to event-based predictions of stock markets (Xie et al. 2013).

However, frame-semantic resources are arguably expensive and time-consuming to build due to difficulties in defining the frames, their granularity, and domain. The complexity of the frame construction and annotation tasks require expertise in the underlying knowledge. Consequently, such resources exist only for a few languages (Boas 2009) and even English is lacking domain-specific frame-based resources. Possible inroads are cross-lingual semantic annotation transfer (Padó and Lapata 2009;

29 We used BabelNet 3.7 extracting all 3,497,327 synsets that were marked as Russian.

30 <https://framenet.icsi.berkeley.edu/fndrupal/luIndex>.

Kidnapping

Definition:

The words in this frame describe situations in which a **Perpetrator** carries off and holds the **Victim** against his or her will by force.

Two men **KIDNAPPED** a Millwall soccer club employee, police said last night.

Not even the **ABDUCTION** of his children **by Captain Hook and his scurvy sidekick, Smee**, can shake Peter's scepticism.

FEs:

Core:

Perpetrator [Perp]	The Perpetrator is the person (or other agent) who carries off and holds the Victim against his or her will.
Semantic Type: Sentient	
Victim [Vict]	The Victim is the person who is carried off and held against his/her will.
Semantic Type: Sentient	

Lexical Units:

abduct.v, abducted.a, abduction.n, abductor.n, kidnap.v, kidnapped.a, kidnapper.n, kidnapping.n, nab.v, shanghai.v, snatch.v, snatcher.n

Figure 8

Definition, examples, core semantic roles, and frame invoking lexical units of the semantic frame “Kidnapping” from the FrameNet resource.

Hartmann, Eckle-Kohler, and Gurevych 2016) or linking FrameNet to other lexical-semantic or ontological resources (Narayanan et al. 2003; Tonelli and Pighin 2009; Laparra and Rigau 2010; Gurevych et al. 2012, *inter alia*). But whereas the arguably simpler task of PropBank-based Semantic Role Labeling has been successfully addressed by unsupervised approaches (Lang and Lapata 2010; Titov and Klementiev 2011), fully unsupervised frame-based semantic annotation exhibits far more challenges, starting with the preliminary step of automatically inducing a set of semantic frame definitions that would drive a subsequent text annotation. We aim at overcoming these issues by automatizing the process of FrameNet construction through unsupervised frame induction techniques using WATSET.

According to our statistics on the dependency-parsed FrameNet corpus of over 150 thousand sentences (Bauer, Fürstenau, and Rambow 2012), the **SUBJ** and **OBJ** relationships are the two most common shortest paths between frame evoking elements (FEEs) and their roles, accounting for 13.5% of instances of a heavy-tail distribution of over 11,000 different paths that occur three times or more in the FrameNet data. Although this might seem a simplification that does not cover prepositional phrases and frames filling the roles of other frames in a nested fashion, we argue that the overall frame inventory can be induced on the basis of this restricted set of constructions, leaving other paths and more complex instances for further work. Thus, we expect the triples obtained from such a Web-scale corpus as DepCC (Panchenko et al. 2018a) to cover most core arguments sufficiently. In contrast to the recent approaches like the one by Jauhar and Hovy (2017), the approach we describe in this section induces semantic frames without any supervision, yet captures only two core roles: the *subject* and the *object* of a frame triggered by *verbal* predicates. Note that it is not generally correct to expect that the SVO triples obtained by a dependency parser are necessarily the core arguments of a predicate. Such roles can be implicit, that is, unexpressed in a given context (Schenk

Table 13

Example of a tricluster of lexical units corresponding to the “Kidnapping” frame from FrameNet.

FrameNet	Role	Lexical Units (LU)
Perpetrator	Subject	kidnapper, alien, militant
FEE	Verb	snatch, kidnap, abduct
Victim	Object	son, people, soldier, child

and Chiarcos 2016), so additional syntactic relationships between frame elements could be taken into account (Kallmeyer, QasemiZadeh, and Cheung 2018).

We cast the frame induction problem as a *triclustering* task (Zhao and Zaki 2005; Ignatov et al. 2015). Triclustering is a generalization of traditional clustering and biclustering problems (Mirkin 1996, page 144), aiming at simultaneously clustering objects along three dimensions, namely, subject, verb, and object in our case (cf. Table 13). First, triclustering allows us to avoid the prevalent pipelined architecture of frame induction approaches, for example, the one by Kawahara, Peterson, and Palmer (2014), where two independent clusterings are needed. Second, benchmarking frame induction as triclustering against other methods on dependency triples makes it possible to abstract away the evaluation of frame induction algorithms from other factors, for example, the input corpus or pre-processing steps, thus allowing a fair comparison of different induction models.

5.1 Frame Induction as a Triclustering Task

We focused on a simple setup for semantic frame induction using two roles and SVO triples, arguing that it still can be useful as frame roles are primarily expressed by subjects and objects, giving rise to semantic structures extracted in an unsupervised way with high coverage. Thus, given a vocabulary V and a set of SVO triples $T \subseteq V^3$ from a syntactically analyzed corpus, our approach for frame induction, called Triframes, constructs a triple graph and clusters it using the WATSET algorithm described in Section 3.

Triframes reduces the frame induction problem to a simpler graph clustering problem. The algorithm has three steps: construction, clustering, and extraction. The triple graph *construction* step, as described in Section 5.1.1, uses a d -dimensional word embedding model $v : V \rightarrow \vec{v} \in \mathbb{R}^d$ to embed triples in a dense vector space for establishing edges between them. The graph *clustering* step, as described in Section 5.1.2, uses a clustering algorithm like WATSET to obtain sets of triples corresponding to the instances of the semantic frames. The final *aggregation* step, as described in Section 5.1.3, transforms the discovered triple clusters into frame-semantic representations. Triframes is parameterized by the number of nearest neighbors $k \in \mathbb{N}$ for establishing edges and a graph clustering algorithm Cluster. The complete pseudocode of Triframes is presented in Algorithm 3.

5.1.1 SVO Triple Similarity Graph Construction. We construct the triple graph $G = (T, E)$ in which the triples are connected to each other according to the semantic similarity of their elements: subjects, verbs, objects. To express similarity, we embed the triples using

Algorithm 3 Unsupervised Semantic Frame Induction from Subject-Verb-Object Triples.

Input: a set of SVO triples $T \subseteq V^3$,
 an embedding model $v \in V \rightarrow \vec{v} \in \mathbb{R}^d$,
 the number of nearest neighbors $k \in \mathbb{N}$,
 a graph clustering algorithm Cluster.

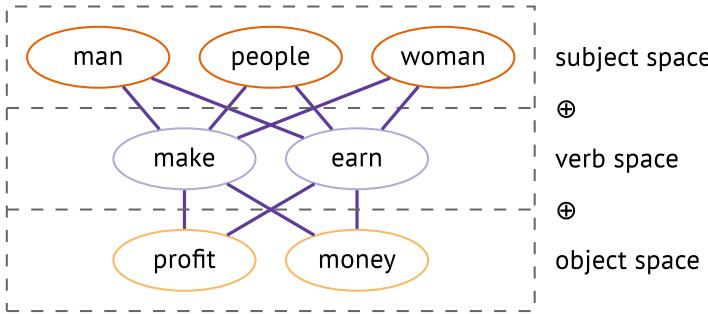
Output: a set of triframe F .

```

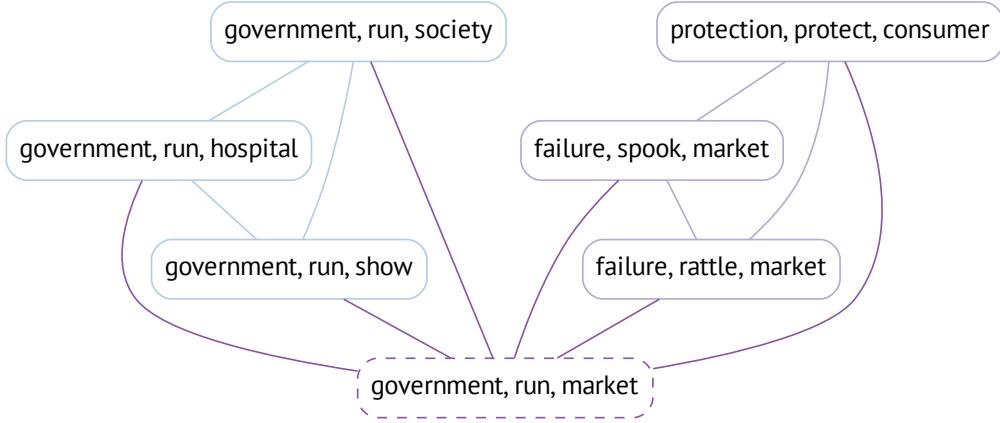
1: for all  $t = (s, p, o) \in T$  do                                 $\triangleright$  Embed the triples
2:    $\vec{t} \leftarrow \vec{s} \oplus \vec{p} \oplus \vec{o}$ 
3:    $E \leftarrow \{(t, t') \in T^2 : t' \in \text{NN}_k(t), t \neq t'\}$      $\triangleright$  Construct edges using nearest neighbors
4:    $G \leftarrow (T, E)$ 
5:    $F \leftarrow \emptyset$ 
6: for all  $C^i \in \text{Cluster}(G)$  do                                 $\triangleright$  Cluster the graph
7:    $f_s \leftarrow \{s \in V : (s, v, o) \in C^i\}$                  $\triangleright$  Aggregate subjects
8:    $f_v \leftarrow \{v \in V : (s, v, o) \in C^i\}$                  $\triangleright$  Aggregate verbs
9:    $f_o \leftarrow \{o \in V : (s, v, o) \in C^i\}$                  $\triangleright$  Aggregate objects
10:   $F \leftarrow F \cup \{(f_s, f_v, f_o)\}$ 
11: return  $F$ 
```

distributional representations of words. In particular, we use a word embedding model to map every triple $t = (s, p, o) \in T$ to a $(3d)$ -dimensional vector $\vec{t} = \vec{s} \oplus \vec{p} \oplus \vec{o}$ (lines 1–2). Such a representation enables computing the distance between the triples as a whole rather than between individual elements of them. The use of distributional models like Skip-Gram (Mikolov et al. 2013) makes it possible to take into account the contextual information of the whole triple. The concatenation of the vectors for words forming triples leads to the creation of a $(|T| \times 3d)$ -dimensional vector space. Figure 9 illustrates this idea: We expect structurally similar triples of different elements to be located in a dense vector space close to each other, and non-similar triples to be located far away from each other.

Given a triple $t \in T$, we denote the $k \in \mathbb{N}$ nearest neighbors extraction procedure of its concatenated embedding from the formed vector space as $\text{NN}_k(t) \subseteq T \setminus \{t\}$. Then, we use the triple embeddings to generate the undirected graph $G = (T, E)$ by constructing the edge set $E \subseteq T^2$. For that, we retrieve k nearest neighbors of each triple vector $\vec{t} \in \mathbb{R}^{3d}$.

**Figure 9**

Concatenation of the vectors corresponding to the triple elements, **subjects**, **verbs**, and **objects**, expresses the structural similarity of the triples.

**Figure 10**

Example of two senses associated with a triple (*government, run, market*).

and establish cosine similarity-weighted edges between the corresponding triples. We establish edges only between the triples appearing in k nearest neighbors (lines 3–4):

$$E = \{(t, t') \in T^2 : t' \in \text{NN}_k(t)\} \quad (12)$$

As a result, the constructed triple graph G has a clustered structure in which the clusters are sets of SVO triples representing the same frame.

5.1.2 Similarity Graph Clustering. We assume that the triples representing similar contexts fill similar roles, which is explicitly encoded by the concatenation of the corresponding vectors of the words constituting the triple (Figure 9). We use the WATSET algorithm to obtain the clustering of the SVO triple graph G (line 6). As described in Section 3, our algorithm treats the SVO triples as the vertices T of the input graph $G = (T, E)$, induces their senses (Figure 10), and constructs an intermediate sense-aware representation that is clustered using a hard clustering algorithm like CW (Biemann 2006). WATSET is a suitable algorithm for this problem because of its performance on the related synset induction task (Section 4), its fuzzy nature, and its ability to find the number of frames automatically.

5.1.3 Aggregating Triframes. Finally, for each cluster $C^i \in C$, we aggregate the subjects, the verbs, and the objects of the contained triples into separate sets (lines 7–9). As a result, each cluster is transformed into a *triframe*, which is a triple that is composed of the subjects $f_s \subseteq V$, the verbs $f_v \subseteq V$, and the objects $f_o \subseteq V$. For example, the triples shown in Figure 9 will form a *triframe* (*{man, people, woman}*, *{make, earn}*, *{profit, money}*).

5.2 Evaluation

Currently, there is no universally accepted approach for evaluating unsupervised frame induction methods. All the previously developed methods were evaluated on completely different incomparable setups and used different input corpora (Titov and

Klementiev 2012; Materna 2013; O’Connor 2013, etc.). We propose a unified methodology by treating the complex multi-stage frame induction task as a straightforward triple clustering task.

5.2.1 Experimental Setup. We compare our method, *Triframes* WATSET, to several available state-of-the-art baselines applicable to our data set of triples (Section 2.3). *LDA-Frames* by Materna (2012, 2013) is a frame induction method based on topic modeling. *Higher-Order Skip-Gram (HOSG)* by Cotterell et al. (2017) generalizes the Skip-Gram model (Mikolov et al. 2013) by extending it from word-context co-occurrence matrices to tensors factorized with a polyadic decomposition. In our case, this tensor consisted of SVO triple counts. *NOAC* by Egurnov, Ignatov, and Mephu Nguifo (2017) is an extension of the Object-Attribute-Condition (*OAC*) triclustering algorithm by Ignatov et al. (2015) to numerically weighted triples. This incremental algorithm searches for dense regions in triadic data. Also, we use five simple baselines. In the *Triadic* baselines, independent word embeddings of subject, object, and verb are concatenated and then clustered using k -means (Hartigan and Wong 1979) and spectral clustering (Shi and Malik 2000). In *Triframes CW*, instead of WATSET, we use CW, a *hard* graph clustering algorithm (Biemann 2006). We also evaluate the performance of Simplified WATSET (Section 3.4). Finally, two trivial baselines are *Singlets* that creates a single cluster per instance and *Whole* that creates one cluster for all elements.

Quality Measure. Following the approach for verb class evaluation by Kawahara, Peterson, and Palmer (2014), we use *normalized modified purity* (nmPU) and *normalized inverse purity* (niPU) as the quality measures for overlapping clusterings. Given the clustering C and the gold clustering C_G , normalized modified purity quantifies the clustering precision as the average of the weighted overlap $\delta_{C^i}(C^i \cap C_G^j)$ between each cluster $C^i \in C$ and the gold cluster $C_G^j \in C_G$, which maximizes the overlap with C^i :

$$\text{nmPU} = \frac{1}{|C|} \sum_{i \in \mathbb{N}: |C^i| > 1}^{|C|} \max_{1 \leq j \leq |C_G|} \delta_{C^i}(C^i \cap C_G^j) \quad (13)$$

where the weighted overlap is the sum of the weights $C^{i,v}$ for each word $v \in C^i$ in i -th cluster: $\delta_{C^i}(C^i \cap C_G^j) = \sum_{v \in C^i \cap C_G^j} C^{i,v}$. Note that nmPU counts all the singleton clusters as wrong. Similarly, normalized inverse purity (collocation) quantifies the clustering recall:

$$\text{niPU} = \frac{1}{|C_G|} \sum_{j=1}^{|C_G|} \max_{1 \leq i \leq |C|} \delta_{C_G^j}(C^i \cap C_G^j) \quad (14)$$

Then, nmPU and niPU are combined together as the harmonic mean to yield the overall clustering F_1 -score, computed as $F_1 = 2 \frac{\text{nmPU} \cdot \text{niPU}}{\text{nmPU} + \text{niPU}}$, which we use to rank the approaches.

Our framework can be extended to the evaluation of more than two roles by generating more roles per frame. Currently, given a set of gold triples generated from the FrameNet, each triple element has a role—for example, “*Victim*,” “*Predator*,” and “*FEE*.” We use a fuzzy clustering evaluation measure that operates not on triples, but instead on a set of tuples. Consider for instance a gold triple (Freddy: *Predator*, kidnap:

FEE, kid: Victim). It will be converted to three pairs (*Freddy, Predator*), (*kidnap, FEE*), (*kid, Victim*). Each cluster in both C and C_G is transformed into a union of all constituent typed pairs. The quality measures are finally calculated between these two sets of tuples corresponding to C and C_G . Note that one can easily pull in more than two core roles by adding to this gold standard set of tuples other roles of the frame, e.g., $\{(forest, Location)\}$. In our experiments, we focused on two main roles as our contribution is related to the application of triclustering methods. However, if more advanced methods of clustering are used, yielding clusters of arbitrary modality (n -clustering), one could also use our evaluation scheme.

Statistical Testing. Because the normalization term of the quality measures used in this experiment does not allow us to compute a contingency table, we cannot directly apply the McNemar's test or a location test to evaluate the statistical significance of the results as we did in our synset induction experiment (Section 4.2.1). Thus, we have applied a bootstrapping approach for statistical significance evaluation as follows. Given a set of clusters C and a set of gold standard clusters C_G , we bootstrap an N -sized distribution of F_1 -scores. On each iteration, we take a sample C' with replacements of $|C|$ elements from C . Then, we compute nmPU, niPU, and F_1 on C' against the gold standard clustering C_G . Finally, for each pair of compared algorithms we use a two-tailed t-test (Welch 1947) from the Apache Commons Math library³¹ to assess the significance of the difference in means between the corresponding bootstrap F_1 -score distributions. Thus, we consider the performance of one algorithm to be higher than the performance of another if both the p-value of the t-test is smaller than the significance level of 0.01 and the mean bootstrap F_1 -score of the first method is larger than that of the second. Because of a high computational complexity of bootstrapping (Dror et al. 2018), we had to limit the value of N to 5,000 in the frame induction experiment and to 10,000 in the verb clustering experiment.

Gold Standard Data Sets. We constructed a gold standard set of triclusters. Each tricluseter corresponds to a FrameNet frame, similarly to the one illustrated in Table 13. We extracted frame annotations from the over 150,000 sentences from FrameNet 1.7 (Baker, Fillmore, and Lowe 1998). We used the frame, FEE, and argument labels in this data set to generate triples in the form $(word_i: role_1, word_j: FEE, word_k: role_2)$, where $word_{i/j/k}$ corresponds to the roles and FEE in the sentence. We omitted roles expressed by multiple words as we use dependency parses, where one node represents a single word only.

For the sentences where more than two roles are present, all possible triples were generated. For instance, consider the sentence "Two *men* kidnapped a soccer club *employee* at the train *station*," where "*men*" has the semantic role of *Perpetrator*, "*employee*" has the semantic role of *Victim*, "*station*" has the semantic role of *Place*, and the word "*kidnapped*" is a frame-evoking lexical element (see Figure 8). In this sentence containing three semantic roles, the following triples will be generated: (*men: Perpetrator, kidn*ap: *FEE, employee: Victim*), (*men: Perpetrator, kidn*ap: *FEE, station: Place*), (*employee: Victim, kidn*ap: *FEE, station: Place*). Sentences with less than two semantic roles were not considered. Finally, for each frame, we selected only two roles that are the most frequently co-occurring in the FrameNet annotated texts. This has left us with about 10^5 instances for the evaluation. For purposes of the evaluation, we operate on the

³¹ <https://commons.apache.org/proper/commons-math/>.

Table 14

Statistics of the evaluation data sets.

Data set	# instances	# unique	# clusters
FrameNet Triples (Bauer et al. 2012)	99,744	94,170	383
Polysemous Verb Classes (Korhonen et al. 2003)	246	110	62

intersection of triples from DepCC and FrameNet. Experimenting on the full set of DepCC triples is only possible for several methods that scale well (WATSET, CW, k -means), but is prohibitively expensive for other methods (LDA-Frames, NOAC) because of the input data size combined with the complexity of these algorithms. During prototyping, we found that removing the triples containing pronouns from both the input and the gold standard data set dramatically reduces the number of instances without the change of ranks in the evaluation results. Thus, we decided to perform our experiments on the whole data set without such a filtering.

In addition to the frame induction evaluation, where subjects, objects, and verbs are evaluated together, we also used a data set of polysemous verb classes introduced by Korhonen, Krymolowski, and Marx (2003) and used by Kawahara, Peterson, and Palmer (2014). Statistics of both data sets are summarized in Table 14. Note that the polysemous verb data set is rather small, whereas the FrameNet triples set is fairly large, enabling reliable comparisons.

Input Data. In our evaluation, we use subject-verb-object triples from the DepCC data set (Panchenko et al. 2018a),³² which is a dependency-parsed version of the Common Crawl corpus, and the standard 300-dimensional Skip-Gram word embedding model trained on Google News corpus (Mikolov et al. 2013). All the evaluated algorithms are executed on the same set of triples, eliminating variations due to different corpora or pre-processing.

5.2.2 Parameter Tuning. We tested various hyper-parameters of each of these algorithms and report the best results overall per frame induction algorithm. We run 500 iterations of the LDA-Frames model with the default parameters (Materna 2013). For HOSG by Cotterell et al. (2017), we trained three vector arrays (for subjects, verbs, and objects) on the 108,073 SVO triples from the *FrameNet* corpus, using the implementation provided by the authors.³³ Training was performed with 5 negative samples, 300-dimensional vectors, and 10 epochs. We constructed an embedding of a triple by concatenating embeddings for subjects, verbs, and objects, and clustered them using k -means with the number of clusters set to 10,000 (this value provided the best performance). We tested several configurations of the NOAC method by Egurnov, Ignatov, and Mephu Nguifo (2017), varying the minimum density of the cluster: The density of 0.25 led to the best results. For our Triframes method, we tried different values of $k \in \{5, 10, 30, 100\}$, while the best results were obtained on $k = 30$ for both Triframes WATSET and CW. Both *Triadic* baselines show the best results on $k = 500$.

³² <https://www.inf.uni-hamburg.de/en/inst/ab/lt/resources/data/depcc.html>.

³³ <https://github.com/azpoliak/skip-gram-tensor>.

Table 15

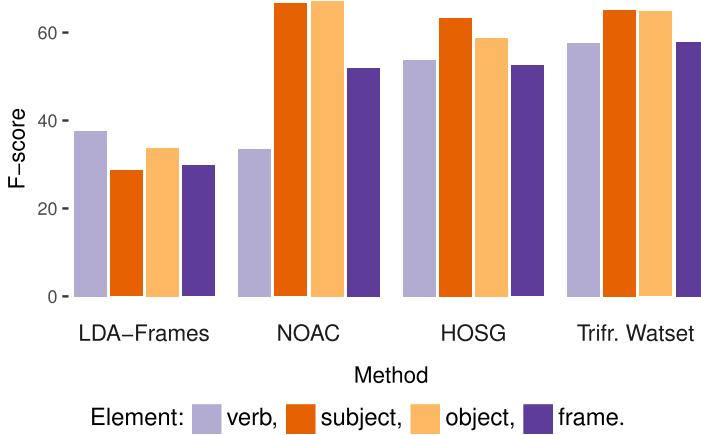
Frame evaluation results on the triples from the FrameNet 1.7 corpus (Baker, Fillmore, and Lowe 1998). The results are sorted by descending order of the Frame F_1 -score. Best results are **boldfaced** and statistically significant results are marked with an asterisk (*). Simplified WATSET is denoted as WATSET\$.

Method	Verb			Subject			Object			Frame		
	nmPU	niPU	F_1									
Triframes WATSET[CW _{top} , CW _{top}]	42.84	88.35	57.70	54.22	81.40	65.09	53.04	83.25	64.80	55.19	60.81	57.87*
Triframes WATSET\$[CW _{top} , CW _{top}]	42.70	87.41	57.37	54.29	78.92	64.33	52.87	83.47	64.74	55.12	59.92	57.42*
Triframes WATSET[MCL, MCL]	52.60	70.07	60.09	55.70	74.51	63.74	54.14	78.70	64.15	60.93	52.44	56.37*
Triframes WATSET\$[MCL, MCL]	55.13	69.58	61.51	55.10	76.02	63.89	54.27	78.48	64.17	60.56	52.16	56.05*
HOSG (Cotterell et al. 2017)	44.41	68.43	53.86	52.84	74.53	61.83	54.73	74.05	62.94	55.74	50.45	52.96
NOAC (Egurnov et al. 2017)	20.73	88.38	33.58	57.00	80.11	66.61	57.32	81.13	67.18	44.01	63.21	51.89*
Triadic Spectral	49.62	24.90	33.15	50.07	41.07	45.13	50.50	41.82	45.75	52.05	28.60	36.91*
Triadic k-Means	63.87	23.16	33.99	63.15	38.20	47.60	63.98	37.43	47.23	63.64	24.11	34.97*
LDA-Frames (Materna 2013)	26.11	66.92	37.56	17.28	83.26	28.62	20.80	90.33	33.81	18.80	71.17	29.75*
Triframes CW	7.75	6.48	7.06	3.70	14.07	5.86	51.91	76.92	61.99	21.67	26.50	23.84
Singltons	0	18.03	0	0	20.56	0	0	17.35	0	81.44	15.50	26.04
Whole	7.35	100.0	13.70	5.62	97.40	10.63	4.24	98.01	8.14	5.07	98.75	9.65

5.2.3 Results and Discussion. We perform two experiments to evaluate our approach: (1) a frame induction experiment on the FrameNet annotated corpus by Bauer, Fürstenau, and Rambow (2012); (2) the polysemous verb clustering experiment on the data set by Korhonen, Krymolowski, and Marx (2003). The first is based on the newly introduced frame induction evaluation scheme (cf. Section 5.2.1). The second one evaluates the quality of verb clusters only on a standard data set from prior work.

Frame Induction Experiment. In Table 15 and Figure 11, the results of the experiment are presented. Triframes based on WATSET clustering outperformed the other methods on both Verb F_1 and overall Frame F_1 . The HOSG-based clustering proved to be the most competitive baseline, yielding decent scores according to all four measures. The NOAC approach captured the frame grouping of slot fillers well but failed to establish good verb clusters. Note that NOAC and HOSG use only the graph of syntactic triples and do not rely on pre-trained word embeddings. This suggests a high complementarity of signals based on distributional similarity and global structure of the triple graph. Finally, the simpler *Triadic* baselines relying on hard clustering algorithms showed low performance, similar to that of *LDA-Frames*, justifying the more elaborate WATSET method. Although we, due to computational reasons (Section 5.2.1), have statistically evaluated only Frame F_1 results, we found all the results except HOSG to be statistically significant ($p \ll 0.01$).

Although triples are intuitively less ambiguous than words, still some frequent and generic triples like (she, make, it) can act as hubs in the graph, making it difficult to split it into semantically plausible clusters. The poor results of the CW hard clustering algorithm illustrate this. Because the hubs are ambiguous (i.e., can belong to multiple clusters), the use of the WATSET fuzzy clustering algorithm that splits the hubs by disambiguating them leads to the best results (see Table 15). We found that on average, WATSET tends to create smaller clusters than its closest competitors, HOSG and NOAC. For instance, an average frame produced by Triframes WATSET[CW_{top}, CW_{top}] has 2.87 ± 4.60 subjects, 3.77 ± 16.31 verbs, and 3.27 ± 6.31 objects. NOAC produced on average 8.95 ± 15.05 subjects, 133.94 ± 227.60 verbs, and 15.17 ± 18.37 objects per frame. HOSG produced on average 3.00 ± 4.20 subjects, 6.49 ± 12.15 verbs, and 2.81 ± 4.89

**Figure 11**

F_1 -score values measured on the FrameNet Corpus (Bauer, Fürstenau, and Rambow 2012). Each block corresponds to the top performance of the method in Table 15.

objects per frame. We conclude that WATSET was producing smaller clusters in general, which appear to be meaningful yet insufficiently coarse-grained, according to the gold standard verb data set used.

Verb Clustering Experiment. Table 16 presents the evaluation results on the second data set for the best models identified in the first data set. The *LDA-Frames* yielded the best results with our approach performing comparably in terms of the F_1 -score. We attribute the low performance of the Triframes method based on CW clustering (Triframes CW) to its hard partitioning output, whereas the evaluation data set contains fuzzy clusters. The simplified version of WATSET has statistically significantly outperformed all other approaches. Although the LDA-Frames algorithm showed a higher value of F_1 than the original version of WATSET in this experiment, we found that its sampled F_1 -score is 44.98 ± 0.04 , while Triframes WATSET[CW_{top}, CW_{top}] showed 47.88 ± 0.01 . Thus, we infer that our method has demonstrated non-significantly lower performance on this verb clustering task. In turn, the NOAC approach showed significantly worse results than both LDA-Frames and our approach ($p \ll 0.01$). Different rankings in Tables 15 and 16 also suggest that frame induction cannot simply be treated as verb clustering and requires a separate task.

Manual Evaluation of the Induced Frames. In addition to the experiments based on gold standard lexical resources, we also performed a manual evaluation. In particular, we assessed the quality of the frames produced by the Triframes WATSET[CW_{top}, CW_{top}] approach using $n = 30$ nearest neighbors for constructing a triple graph, which showed the best performance during automatic evaluation (Tables 15 and 16).

To prepare the data for a manual annotation, we sampled 100 random frames and manually annotated them with three different annotators. For the convenience of the annotators, before drawing a sample we removed pronouns and prepositions from the frame elements while keeping them containing at least two different lexical units. This is to remove rather meaningful triples, for example, (her, make, it), which are, however, present in large amounts in the FrameNet gold standard data set.

Table 16

Evaluation results on the data set of polysemous verb classes by Korhonen, Krymolowski, and Marx (2003). The results are sorted by the descending order of F_1 -score. Best results are **boldfaced** and statistically significant results are marked with an asterisk (*). Simplified WATSET is denoted as WATSET\$.

Method	nmPU	niPU	F_1
Triframes WATSET\$[CW _{top} , CW _{top}]	41.21	62.82	49.77*
LDA-Frames (Materna 2013)	52.60	45.84	48.98
Triframes WATSET[CW _{top} , CW _{top}]	40.05	62.09	48.69*
NOAC (Egurnov et al. 2017)	36.43	63.68	46.35*
Triframes WATSET[MCL, MCL]	39.26	54.92	45.78*
Triframes WATSET\$[MCL, MCL]	36.31	53.81	43.36*
Triadic Spectral	45.70	38.96	42.06
HOSG (Cotterell et al. 2017)	38.22	43.76	40.80*
Triadic <i>k</i> -means	46.76	28.92	35.74*
Triframes CW	18.05	12.72	14.92
Whole	24.14	79.09	36.99
Singletons	0	27.21	0

# 1268	Subjects:	expert, scientist, lecturer, engineer, analyst
	Verbs:	study, examine, tell, detect, investigate, do, observe, hold, find, have, predict, claim, notice, give, discover, explore, learn, monitor, check, recognize, demand, look, call, engage, spot, inspect, ask
	Objects:	view, problem, gas, area, change, market
# 1378	Subjects:	leader, officer, khan, president, government, member, minister, chief, chairman
	Verbs:	belong, run, head, spearhead, lead
	Objects:	party, people
# 4211	Subjects:	evidence, research, report, survey
	Verbs:	prove, reveal, tell, show, suggest, confirm, indicate, demonstrate
	Objects:	method, evidence

Figure 12

Examples of “good” frames produced by the Triframes WATSET[CW_{top}, CW_{top}] method as labeled by our annotators; frame identifiers are present in the first column; pronouns and prepositions are omitted.

In this study, annotators were instructed to annotate a frame as “good” if its elements (SVO) generally make sense together and each element is a reasonable set of lexical units. In total, the annotators judged 63 frames out of 100 to be good with a Fleiss (1971) κ agreement of 0.816.³⁴ Although this is a rather general definition, the high agreement rate seems to suggest that it still provides a meaningful definition shared across annotators. Figure 12 presents examples of “good” frames, that is, those which

³⁴ We used the DKPro Agreement toolkit by Meyer et al. (2014) to compute the inter-annotator agreement.

# 8	Subjects: wine, act, power Verbs: hearten, bring, discourage, encumber, ... 432 more verbs..., build, chew, unsettle, snap Objects: right, good, school, there, thousand
# 1057	Subjects: parent, scientist, officer, event Verbs: promise, pledge Objects: parent, be, good, government, client, minister, people, coach
# 1657	Subjects: people, doctor Verbs: spell, steal, tell, say, know Objects: egg, food, potato

Figure 13

Examples of “bad” frames produced by the Triframes WATSET[CW_{top}, CW_{top}] method as labeled by our annotators; frame identifiers are present in the first column, pronouns and prepositions are omitted.

are labeled as semantically plausible by our annotators. Figure 13 shows examples of “bad” frames according to the same criteria. These frames are available for download.³⁵

6. Application to Unsupervised Distributional Semantic Class Induction

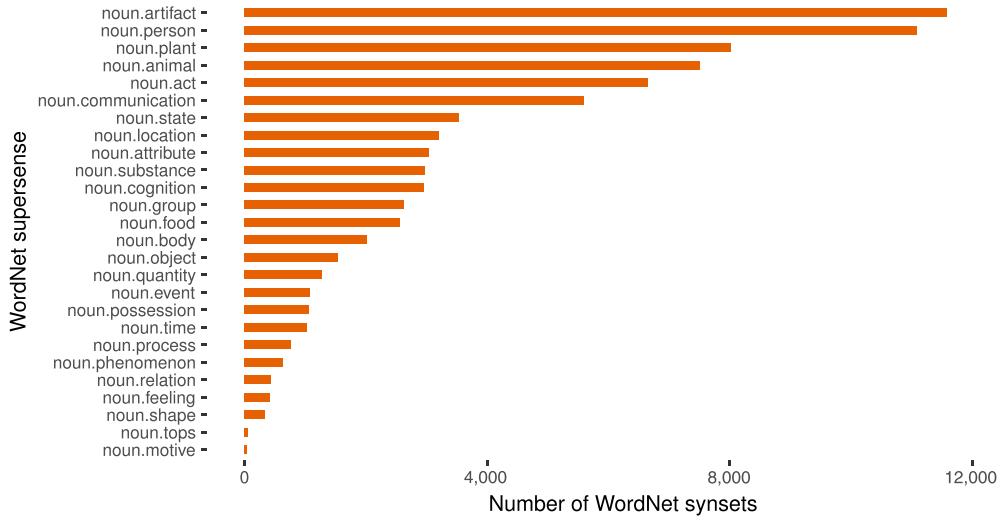
In this section, we investigate the applicability of our graph clustering technique in another unsupervised resource induction task. The first two experiments investigated the acquisition of two linguistic symbolic structures from two different types of graphs—namely, synsets induced from graphs of synonyms (Section 4) and semantic frames induced from graphs of distributionally related syntactic triples (Section 5). In this section, we show how WATSET can be used to induce a third type of structure, namely, *semantic classes* from a graph of distributionally related words, also known as a *distributional thesaurus* (or DT) (see Lin 1998; Biermann and Riedl 2013). In the context of this article, *semantic classes* will be considered as semantically plausible groups of words or word senses that have some common semantic feature.

The following sections will provide details of this experiment. In particular, Section 6.1 presents two data sets that are used as gold standard clustering in the experiments. Section 6.2 presents the input graphs that are clustered using our approach to induce semantic structure. Finally, in Section 6.3 results of the experiments are presented and discussed comparing them to the baseline clustering algorithms.

6.1 Semantic Classes in Lexical Semantic Resources

A *semantic class* is a set of words that share the same semantic feature (Kozareva, Riloff, and Hovy 2008). Depending on the definition of the notion of the *semantic feature*, the granularity and sizes of semantic classes may vary greatly. Examples of concrete semantic classes include sets of animals (dog, cat, ...), vehicles (car, motorcycle, ...), and fruit trees (apple tree, peach tree, ...). In this experiment, we use a gold standard derived from a reference lexicographical database, namely, WordNet (Fellbaum 1998).

³⁵ The examples are from the file `triv2v-watset-n30-top-top-triples.txt` is available in the “Downloads” section of our GitHub repository at <https://github.com/uhh-lt/triframes>.

**Figure 14**

A summary of the noun semantic classes in WordNet supersenses (Ciaramita and Johnson 2003).

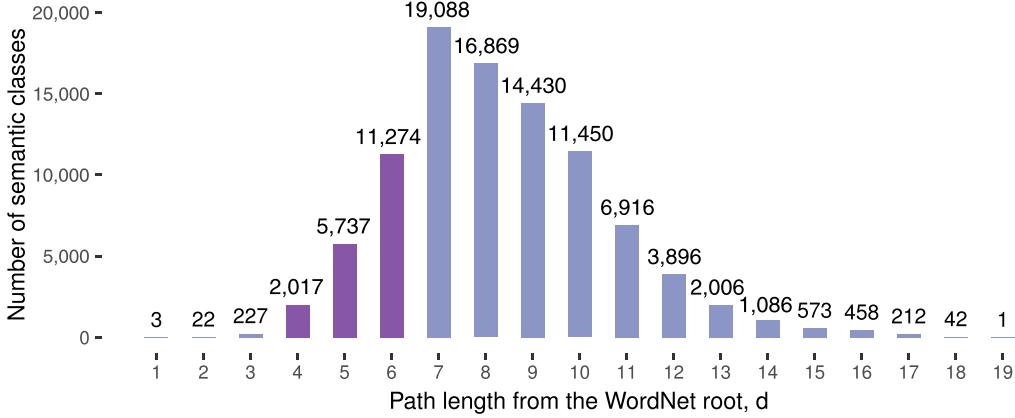
This allows us to benchmark the ability of WATSET to reconstruct the semantic lexicon of such a reliable reference resource that has been widely used in NLP for many decades.

6.1.1 WordNet Supersenses. The first data set used in our experiments consists of 26 broad semantic classes, also known as **supersenses** in the literature (Ciaramita and Johnson 2003): *person, communication, artifact, act, group, food, cognition, possession, location, substance, state, time, attribute, object, process, tops, phenomenon, event, quantity, motive, animal, body, feeling, shape, plant, and relation*.

This system of broad semantic categories was used by lexicographers who originally constructed WordNet to thematically order the synsets; Figure 14 shows the distribution of the 82,115 noun synsets from WordNet 3.1 across the supersenses. In our experiments in this section, these classes are used as gold standard clustering of word senses as recorded in WordNet. One can observe a Zipfian-like power-law (Zipf 1949) distribution with a few clusters, such as *artifact* and *person*, accounting for a large fraction of all nouns in the resource. Overall, in this experiment we decided to focus on nouns, as the input distributional thesauri used in this experiment (as presented in Section 6.2) are most studied for modeling of noun semantics (Panchenko et al. 2016b).

The WordNet supersenses were applied later also for word sense disambiguation as a system of broad sense labels (Flekova and Gurevych 2016). For BabelNet, there is a similar data set called BabelDomains (Camacho-Collados and Navigli 2017) produced by automatically labeling BabelNet synsets with 32 different domains based on the topics of Wikipedia featured articles. Despite the larger size, however, BabelDomains provides only a silver standard (being semi-automatically created). We thus opt in the following to use WordNet supersenses only, because they provide instead a gold standard created by human experts.

6.1.2 Flat Cuts of the WordNet Taxonomy. The second type of semantic classes used in our study are more semantically specific and defined as subtrees of WordNet at some fixed

**Figure 15**

Relationship between the number of semantic classes and path length from the WordNet (Fellbaum 1998) root. We have chosen $d \in \{4, 5, 6\}$ for our experiments.

path length of d steps from the root node. We used the following procedure to gather these semantic classes.

First, we find a set of synsets that are located an exact distance of d edges from the root node. Each such starting node (e.g., the synset `plant_material.n.01`) identifies one semantic class. This starting node and all its descendants (e.g., `cork.n.01`, `coca.n.03`, `ethyl_alcohol.n.1`, `methylated_spirit.n.01`, and so on, in the case of the *plant material* example) are included in the semantic class. Finally, we remove semantic classes that contain only one element as our goal is to create a gold standard data set for clustering. Figure 15 illustrates distribution of the number of semantic classes as a function of the path length from the root. As one may observe, the largest number of clusters is obtained for the path length d of 7. In our experiments, we use three versions of these WordNet “taxonomy cuts,” which correspond to $d \in \{4, 5, 6\}$, because the cluster sizes generated at these levels are already substantially larger than those from the supersense data set while providing a complementary evaluation at different levels of granularities. Although at some levels, such as $d = 2$, the number of semantic classes is similar to the number of supersenses (Ciaramita and Johnson 2003), there is no one-to-one relationship between them. As Richardson, Smeaton, and Murphy (1994) point out, this cut-based derivative resource might bias toward the concepts belonging to shallow hierarchies: the node for “horse” is 10 levels from the root, whereas the node for “cow” is 13 levels deep. However, we believe that it adds an additional perspective to our evaluation while keeping the interpretability at the same time. Examples of the extracted semantic classes are presented in Table 17.

6.2 Construction of a Distributional Thesaurus

A distributional thesaurus (Lin 1998) is an undirected graph of semantically related words, with edges such as {Python, Perl}. We base our approach on the distributional hypothesis (Firth 1957; Turney and Pantel 2010; Clark 2015) to generate graphs of semantically related words for this experiment. The graphs represent k nearest neighboring words that are semantically related to each other in a vector space. More

Table 17

Examples of semantic classes extracted from WordNet hierarchy of synsets for the path length $d = 5$ from the root synset.

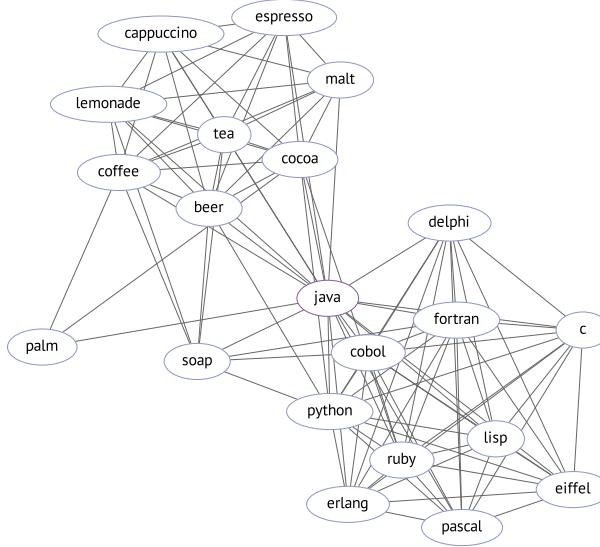
Root Synset	Child Synsets
rock.n.02	aphanite.n.01, caliche.n.02, claystone.n.01, dolomite.n.01, emery_stone.n.01, fieldstone.n.01, gravel.n.01, ballast.n.02, bank_gravel.n.01, shingle.n.02, greisen.n.01, igneous_rock.n.01, adesite.n.01, andesite.n.01, ... 63 more entries..., tufa.n.01
toxin.n.01	animal_toxin.n.01, venom.n.01, kokoi_venom.n.01, snake_venom.n.01, anatoxin.n.01, botulin.n.01, cytotoxin.n.01, enterotoxin.n.01, nephrotoxin.n.01, endotoxin.n.01, exotoxin.n.01, ... 19 more entries..., ricin.n.01
axis.n.01	coordinate_axis.n.01, x-axis.n.01, y-axis.n.01, z-axis.n.01, major_axis.n.01, minor_axis.n.01, optic_axis.n.01, principal_axis.n.01, semimajor_axis.n.01, semiminor_axis.n.01

specifically, the dimensions of the vector space represent salient syntactic dependencies of each word extracted using a dependency parser. For this, we use the JoBimText framework for computation of count-based distributional models from raw text collections (Biemann and Riedl 2013).³⁶ Although similar graphs could be derived also from neural distributional models, such as Word2Vec (Mikolov et al. 2013), it was shown in Riedl (2016) and Riedl and Biemann (2017) that the quality of syntactically-based graphs is generally superior.

The JoBimText framework involves several steps. First, it takes an unlabeled input text corpus and performs dependency parsing so as to extract features representing each word. Each word is represented by a bag of syntactic dependencies such as `conj_and(Ruby, ·)` or `prep_in(code, ·)`, extracted from the dependencies of MaltParser (Nivre, Hall, and Nilsson 2006), which are further collapsed using the tool by Ruppert et al. (2015) in the notation of Stanford Dependencies (de Marneffe, MacCartney, and Manning 2006).

Next, semantically related words are computed for each word in the input corpus. Features of each word are weighted and ranked using the Local Mutual Information measure (Evert 2005). Subsequently, these word representations are pruned, keeping 1,000 most salient features per word (`fpw`) and 1,000 most salient words per feature (`wpf`), where `fpw` and `wpf` are the parameters specific to the JoBimText framework. The pruning reduces computational complexity and noise. Finally, word similarities are computed as the number of common features for two words. This is, again, followed by a pruning step in which for every word, only the k of 200 most similar terms are kept. The ensemble of all of these words is the distributional thesaurus, which is used in the following experiments. Note that each word in such a thesaurus (i.e., a graph of semantically related words) is potentially ambiguous.

36 <http://www.jobimtext.org>.

**Figure 16**

An example of the lexical unit “*java*” and a part of its neighborhood in a distributional thesaurus. This polysemous word is not disambiguated, so it acts as a hub between two different senses.

The last stage of the JoBimText approach performs induction of senses, although here we do not use output of this stage, but instead apply the WATSET algorithm to the distributional thesaurus with ambiguous word entries. The process of computation of a distributional thesaurus using the JoBimText framework is described in greater detail in Biemann et al. (2018, Section 4.1).

As an input corpus, we use a text collection of about 9.3 billion tokens that consists of a concatenation of Wikipedia,³⁷ ukWaC (Ferraresi et al. 2008), Gigaword (Graff and Cieri 2003), and LCC (Richter et al. 2006) corpora. Given the large size of these corpora, the graphs are built using an implementation of the JoBimText framework in Apache Spark,³⁸ which enables efficient distributed computation of large text collection on a distributed computational cluster.³⁹

Figure 16 shows an example from the obtained distributional thesaurus. As in the experiments described in Sections 4 and 5, we assume that polysemous nodes serve as hubs that connect different unrelated clusters.

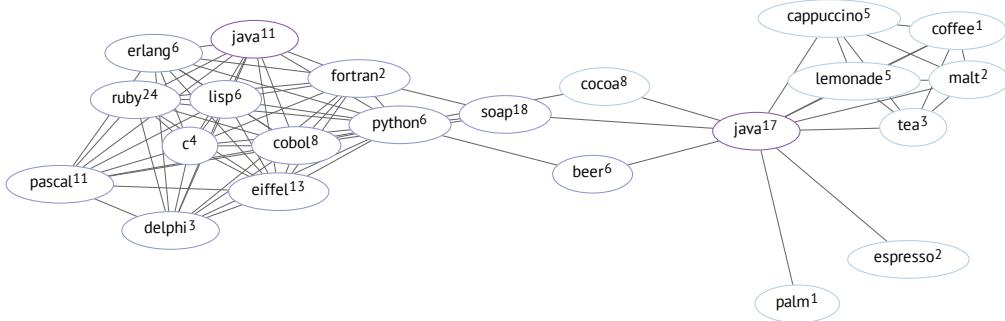
6.3 Evaluation

We cast the semantic class induction problem as a task of clustering distributionally related graphs of words and word senses, which is conceptually similar to our synset induction task in Section 4. Figure 17 shows an example of the sense graph (Section 3.3) built by WATSET before running a global clustering algorithm, which induces the sense-aware semantic classes based on the distributional thesaurus example in Figure 16.

³⁷ <https://doi.org/10.5281/zenodo.229904>.

³⁸ <https://spark.apache.org>.

³⁹ <https://github.com/uhh-lt/josimtext>.

**Figure 17**

An example of the *sense graph* built by WATSET for two senses of the lexical unit “java” using CW_{log} for local clustering. In contrast to Figure 16, in this *disambiguated* distributional thesaurus the node corresponding to the lexical unit “java” is split: $java^{11}$ is connected to [programming languages](#) and $java^{17}$ is connected to [drinks](#).

6.3.1 Experimental Set-Up. Similarly to our synset induction experiment (Section 4.2.1), we study the performance of clustering algorithms by comparing the clustering of the same input distributional thesaurus to a gold standard clustering. We used the same implementations and algorithms as all other experiments reported in this paper, such as MCL by van Dongen (2000), CW by Biemann (2006), and MaxMax (Hope and Keller 2013a). We did not evaluate such algorithms as CPM (Palla et al. 2005) and ECO (Gonçalo Oliveira and Gomes 2014) because of their poor performance shown on the synset induction task.

Input Data. We use the distributional thesaurus as described in Section 6.2. Because the original distributional thesaurus graph has approximately 600 million edges, we pruned it by removing all the edges having the minimal weight (i.e., 0.001 in our case). Also, because of the difference in lexicons between the gold standards and the input graph, we performed additional pruning by removing all the edges connecting words missing the gold standard lexicons. As a result, we obtained four different pruned input graphs (Table 18). We performed no parameter tuning in this experiment, so we report the best-performing configuration of each method among other ones.

Gold Standard. We use two different kinds of semantic classes for evaluation purposes. Both of the semantic class types used are based on the WordNet lexical database

Table 18

Properties of the input data sets used in the semantic class induction experiment compared with the original distributional thesaurus (DT) by Biemann and Riedl (2013).

DT Pruning Method	# of nodes	# of edges
Unpruned (Biemann and Riedl 2013)	4,430,170	595,916,414
Supersenses (Ciaramita 2003)	37,937	6,944,731
Path Length of $d = 4$	33,213	5,841,359
Path Length of $d = 5$	32,048	5,478,110
Path Length of $d = 6$	29,515	4,814,132

(Fellbaum 1998) yet they have widely different granularities. First, we use the WordNet supersenses data set by Ciaramita and Johnson (2003). Second, we use our path-based gold standards of lengths 4, 5, and 6, as described in Section 6.1.

Quality Measure. In the synset induction experiment (Section 4.2.1) we use the pairwise F_1 -score (Manandhar et al. 2010) as the performance indicator. However, because the average size of a cluster in this experiment is much higher (Table 18 and Figure 14), we found that the enumeration of 2-combinations of semantic class elements is not computationally tractable in reasonable time on relatively large data sets like the ones we use in this experiment. For example, a cluster of 10,000 elements needs to be transformed into a sufficiently large set of $\frac{1}{2} \times 10^5 \times (10^5 - 1) \approx 5 \times 10^9$ pairs, which is inconvenient for processing. Therefore, we used the same quality measure as in our unsupervised lexical semantic frame induction experiment (Section 5.2.1), namely, normalized modified purity (nmPU), and normalized inverse purity (niPU), as defined by Kawahara, Peterson, and Palmer (2014).

Statistical Testing. Because the chosen quality measure does not allow the computation of a contingency table, we use exactly the same procedure for statistical testing as in the experiment on lexical semantic frame induction (Section 5.2.1). Due to a high computational complexity of the bootstrapping statistical testing procedure (Dror et al. 2018), we limited the number of samples N to 5,000 in this experiment.

6.3.2 Results and Discussion.

Comparison to Baselines. Table 19 shows the evaluation results on the WordNet supersenses data set. We found that our approach, WATSET[CW_{lin}, CW_{log}], shows statistically significantly better results with respect to F_1 -score ($p \ll 0.01$) than all the methods, apart from Simplified WATSET in the same configuration. The experimental results in Table 20 obtained on different variations of our WordNet-based gold standard, as described in Section 6.1, confirm a high performance of WATSET on all the evaluation data sets. Thus, results of experiments on these four types of semantic classes of greatly variable granularity (from 26 classes for the supersenses to 11,274 classes for the flat cut with $d = 6$) lead to similar conclusions about the advantage of the WATSET approach, as compared to the baseline clustering algorithms.

Table 21 shows examples of the obtained semantic classes of various sizes for the best WATSET configuration on the WordNet supersenses data set. During error analysis

Table 19

Comparison of the graph clustering methods against the WordNet supersenses data set by Ciaramita and Johnson (2003); best configurations of each method in terms of F_1 -scores are shown. Results are sorted by F_1 -score; top values of each measure are **boldfaced**, and statistically significant results are marked with an asterisk (*). Simplified WATSET is denoted as WATSET\$.

Method	# clusters	nmPU	niPU	F_1
WATSET[CW _{lin} , CW _{log}]	47,054	57.20	40.52	47.44
WATSET\$[CW _{lin} , CW _{log}]	47,797	58.16	39.86	47.30*
CW _{log}	108	35.03	46.17	39.84*
MCL	368	61.34	15.31	24.50*
MaxMax	4,050	68.48	4.15	7.82

Table 20

Evaluation results on path-limited versions of WordNet by 4, 5, and 6; best configurations of each method in terms of F_1 -scores are shown. Results are sorted by F_1 -score on the $d = 6$ WordNet slice; top values of each measure are **boldfaced**. Simplified WATSET is denoted as WATSET\$.

Method	$d = 4$			$d = 5$			$d = 6$		
	nmPU	niPU	F_1	nmPU	niPU	F_1	nmPU	niPU	F_1
WATSET\$[CW _{lin} , CW _{top}]	47.43	42.63	44.90	45.26	42.67	43.93	40.20	44.37	42.18
WATSET[CW _{lin} , CW _{top}]	47.38	42.65	44.89	44.86	43.03	43.93	40.07	44.14	42.01
CW _{lin}	34.09	40.98	37.22	34.92	40.65	37.57	31.84	41.89	36.18
CW _{log}	29.00	44.85	35.23	29.63	44.72	35.64	26.00	46.36	33.31
MCL	54.90	19.63	28.92	45.32	22.59	30.15	38.38	26.96	31.67
MaxMax	59.29	6.93	12.42	52.65	10.14	17.01	47.28	13.69	21.23

Table 21

Sample semantic classes induced by the WATSET[CW_{lin}, CW_{log}] method, according to the WordNet supersenses data set by Ciaramita and Johnson (2003).

Size	Semantic Class
7	dye, switch-hitter, dimaggio, hitter, gwynn, three-hitter, muser
13	worm, octopus, pike, anguillidae, congridae, conger, anguilliformes, eel, marine, grouper, muraenidae, moray, elver
16	gothic, excelsior, roman, microgramma, stymie, dingbat, italic, century, trajan, outline, twentieth, bodoni, serif, lydian, headline, goudy
20	nickel, steel, alloy, chrome, titanium, cent, farthing, cobalt, brass, denomination, fineness, paisa, copperware, dime, cupronickel, centavo, avo, threepence, coin, centime
23	prochlorperazine, nicotine, tadalafil, billionth, ricin, pravastatin, multivitamin, milligram, anticoagulation, carcinogen, microgram, niacin, l-dopa, lowering, arsenic, morphine, nevirapine, caffeine, ritonavir, aspirin, neostigmine, rem, milliwatt
54	integer, calculus, theta, pyx, curvature, saturation, predicate, ... 40 more words..., viscosity, brightness, variance, lattice, polynomial, rho, determinant
369	electronics, siren, dinky, banjo, luo, shawm, shaker, helicon, rhodes, conducting, ... 349 more words..., narrator, paradiddle, clavichord, chord, consonance, sextet, zither, cantor, viscera, axiom
1,093	egg, pinworm, forager, decidua, psittacus, chimera, coursing, silkworm, spirochete, radicle, ... 1073 more words..., earthworm, annelida, integument, pisum, biter, wilt, heartwood, shellfish, swarm, cryptomonad

we found two primary causes of errors: incorrectly identified edges and overly specific sense contexts.

Because we performed only a minimal pruning of the input distributional thesaurus, this contains many edges with low weights that typically represent mistakenly recognized relationships between words. Such edges, when appearing between two disjoint meaningful clusters, act as hubs, which WATSET puts in both clusters. For example, a sense graph in Figure 17 has a node *soap*¹⁸ incorrectly connected to a drinks-related node *java*¹⁷ instead of the node *java*¹¹, which is more related to programming languages.⁴⁰ Reliable distinction between “legitimate” polysemous nodes and incorrectly placed hubs is a direction for future work.

The node sense induction approach of WATSET, as described in Section 2.2, takes into account only the neighborhood of the target node, which is a first-order ego network (Everett and Borgatti 2005). As we observe throughout all the experiments in this article, WATSET tends to produce more fine-grained senses than one might expect. These fine-grained senses, in turn, lead to the global clustering algorithm to include incoherent nodes to clusters as in Table 21. We believe that taking into account additional features, such as second-order ego networks, to induce coarse-grained senses could potentially improve the overall performance of our algorithm (at a higher computational cost).

We found a generally poor performance of MCL in this experiment due to its tendency to produce fine-grained clusters by isolating hubs from their neighborhoods. Although this behavior improved the results on the synset induction task (Section 4.2.3), our distributional thesaurus is a more complex resource as it expresses semantic relationships other than synonymy, so the incorrectly identified edges affect MCL as well as WATSET.

Impact of Distributional Thesaurus Pruning on Ambiguity. In order to study the effect of pruning, we performed another experiment on a DT that was pruned using a relatively high edge weight threshold of 0.01, which is 10 times larger than the minimal threshold we used in the experiment described in Section 6.3. A manual inspection of the pruned graph showed that most, if not all, nodes were either monosemous words or proper nouns, so hard clustering algorithms should have an advantage in this scenario. Table 22 confirms that in this setup soft clustering algorithms, such as WATSET and MaxMax, are clearly outperformed by hard clustering algorithms, which are more suitable for processing monosemous word graphs. Because our algorithm explicitly performs node sense induction to produce fine-grained clusters, we found that an average semantic class produced by WATSET[CW_{top}, CW_{top}] has 10.77 ± 187.37 words, whereas CW_{log} produced semantic classes of $133.46 \pm 1,317.97$ words on average.

To summarize, in contrast with synonymy dictionaries, whose completeness and availability are limited (Section 4.2.3), a distributional thesaurus can be constructed for any language provided with a relatively large text corpus. However, we found that they need to be carefully pruned to reduce the error rate of clustering algorithms (Panchenko et al. 2018b).

⁴⁰ Strictly speaking, SOAP (Simple Object Access Protocol) is not a programming language, so the presence of this node in the graphs demonstrated in Figures 16 and 17 is a mistake.

Table 22

Comparison of the graph clustering methods on the pruned DT with an edge threshold of 0.01 against the WordNet supersenses data set by Ciaramita and Johnson (2003); best configurations of each method in terms of F_1 -scores are shown. Results are sorted by F_1 -score; top values of each measure are **boldfaced**. Simplified WATSET is denoted as WATSET\$.

Method	# clusters	nmPU	niPU	F_1
CW _{log}	183	39.72	28.46	33.16
WATSET\$[CW _{top} , CW _{top}]	3,944	57.22	20.21	29.87
WATSET[CW _{top} , CW _{top}]	3,954	57.38	19.91	29.56
MCL	526	65.12	8.46	14.98
MaxMax	3,761	72.71	2.00	3.88

7. Conclusion

In this article, we presented WATSET, a generic meta-algorithm for fuzzy graph clustering. This algorithm creates an intermediate representation of the input graph that naturally reflects the “ambiguity” of its nodes. Then, it uses hard clustering to discover clusters in this “disambiguated” intermediate graph. This enables straightforward semantic-aware grouping of relevant objects together. We refer to WATSET as a meta-algorithm because it does not perform graph clustering per se. Instead, it encapsulates the existing clustering algorithms and builds a sense-aware representation of the input graph, which we call a *sense graph*. Although we use the sense graph in this article exclusively for clustering, we believe that it can be useful for more applications.

The experiments show that our algorithm performs fuzzy graph clustering with a high accuracy. This is empirically confirmed by successfully applying WATSET to complex language processing, including tasks like unsupervised induction of synsets from a synonymy graph, semantic frames from dependency triples, as well as semantic class induction from a distributional thesaurus. In all cases, the algorithm successfully handled the ambiguity of underlying linguistic objects, yielding the state-of-the-art results in the respective tasks. WATSET is computationally tractable and its local steps can easily be run in parallel.

As future work we plan to apply WATSET to other types of linguistic networks to address more natural language processing tasks, such as taxonomy induction based on networks of noisy hypernyms extracted from text (Panchenko et al. 2016a). Additionally, an interesting future challenge is the development of a scalable graph clustering algorithm that can natively run in a parallel distributed manner (e.g., on a large distributed computational cluster). The currently available algorithms, such as MCL (van Dongen 2000) and CW (Biemann 2006), cannot be trivially implemented in such a fully distributed environment, limiting the scale of language graph they can be applied to. Another direction of future work is using WATSET in downstream applications. We believe that our algorithm can successfully detect structure in a wide range of different linguistic and non-linguistic data sets, which can help in processing out-of-vocabulary items or resource-poor languages or domains without explicit supervision.

Implementation. We offer an efficient open source multi-threaded implementation of WATSET (Algorithm 1) in the Java programming language.⁴¹ It uses a thread pool

41 <https://github.com/nlpub/watset-java>.

to simultaneously perform *local* steps, such as node sense induction (lines 1–9, one word per thread) and context disambiguation (lines 11–15, one sense per thread). Our implementation includes Simplified WATSET (Algorithm 2) and also features both a command-line interface and an application programming interface for integration into other graph and language processing pipelines in a generic way. Additionally, we bundle with it our own implementations of Markov Clustering (van Dongen 2000), Chinese Whispers (Biemann 2006), and MaxMax (Hope and Keller 2013a) algorithms. Also, we offer an implementation of the Triframes frame induction approach⁴² and an implementation of the semantic class induction approach.⁴³ The data sets produced during this study are available on Zenodo.⁴⁴

Acknowledgments

We acknowledge the support of the Deutsche Forschungsgemeinschaft (DFG) foundation under the “JOIN-T” and “ACQuA” projects, the Deutscher Akademischer Austauschdienst (DAAD), and the Russian Foundation for Basic Research (RFBR) under the project no. 16-37-00354 мол_а. We also thank Andrew Krizhanovsky for providing a parsed Wiktionary, Natalia Loukachevitch for the provided RuWordNet data set, Mikhail Chernoskutov for early discussions on computational complexity of WATSET, and Denis Shirgin, who actually suggested the WATSET name. Furthermore, we thank Dmitry Egurnov, Dmitry Ignatov, and Dmitry Gnatyshak for help in operating the NOAC method using the multimodal clustering toolbox. We are grateful to Ryan Cotterell and Adam Poliak for a discussion and an implementation of the High-Order Skip Gram (HOSG) method. We thank Bonaventura Coppolla for discussions and preliminary work on graph-based frame induction and Andrei Kutuzov, who conducted experiments with the HOSG-based baseline related to the frame induction experiment. We thank Stefano Faralli for early work on graph-based sense disambiguation. We thank Rotem Dror for discussion of the theoretical background underpinning the statistical testing approach that we use in this article. We are grateful to Federico Nanni and Gregor Wiedemann for proofreading this article. Finally, we thank three anonymous reviewers for insightful comments on the present article.

References

- Abramov, Nikolay. 1999. Словарь русских синонимов и сходных по смыслу выражений [*The Dictionary of Russian Synonyms and Semantically Related Expressions*], 7th edition. Русские словари [Russian Dictionaries], Moscow. In Russian.
- Apidianaki, Marianna and Benoît Sagot. 2014. Data-driven synset induction and disambiguation for wordnet development. *Language Resources and Evaluation*, 48(4):655–677.
- Baker, Collin F., Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, pages 86–90, Montréal.
- Bartunov, Sergey, Dmitry Kondrashkin, Anton Osokin, and Dmitry P. Vetrov. 2016. Breaking sticks and ambiguities with adaptive skip-gram. *Journal of Machine Learning Research*, 51:130–138.
- Bauer, Daniel, Hagen Fürstenau, and Owen Rambow. 2012. The dependency-parsed Framenet corpus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 3861–3867, Istanbul.
- Ben Aharon, Roni, Idan Szpektor, and Ido Dagan. 2010. Generating entailment rules from FrameNet. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 241–246, Uppsala.

⁴² <https://github.com/uhh-lt/triframes>.

⁴³ <https://github.com/umanlp/watset-classes>.

⁴⁴ <https://doi.org/10.5281/zenodo.2621579>.

- Biemann, Chris. 2006. Chinese Whispers: An efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*, 73–80, New York, NY.
- Biemann, Chris. 2012. *Structure Discovery in Natural Language*. Theory and Applications of Natural Language Processing. Springer Berlin Heidelberg.
- Biemann, Chris, Stefano Faralli, Alexander Panchenko, and Simone Paolo Ponzetto. 2018. A framework for enriching lexical semantic resources with distributional semantics. *Natural Language Engineering*, 24(2):265–312.
- Biemann, Chris and Martin Riedl. 2013. Text: Now in 2D! A framework for lexical expansion with contextual similarity. *Journal of Language Modelling*, 1(1):55–95.
- Bizer, Christian, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. DBpedia—A crystallization point for the Web of Data. *Journal of Web Semantics*, 7(3):154–165.
- Blei, David M., Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.
- Blondel, Vincent D., Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.
- Boas, Hans C. 2009. *Multilingual FrameNets in Computational Lexicography: Methods and Applications*. Trends in Linguistics. Studies and Monographs. Mouton de Gruyter.
- Braslavski, Pavel, Dmitry Ustalov, Mukhin Mukhin, and Yuri Kiselev. 2016. YARN: Spinning-in-progress. In *Proceedings of the 8th Global WordNet Conference*, pages 58–65, Bucharest.
- Burchardt, Aljoscha, Marco Pennacchiotti, Stefan Thater, and Manfred Pinkal. 2009. Assessing the impact of frame semantics on textual entailment. *Natural Language Engineering*, 15(4):527–550.
- Camacho-Collados, Jose and Roberto Navigli. 2017. BabelDomains: Large-scale domain labeling of lexical resources. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 223–228, Valencia.
- Cecchini, Flavio Massimiliano, Martin Riedl, Elisabetta Fersini, and Chris Biemann. 2018. A comparison of graph-based word sense induction clustering algorithms in a pseudoword evaluation framework. *Language Resources and Evaluation*, 733–770.
- Cheung, Jackie C. K., Hoifung Poon, and Lucy Vanderwende. 2013. Probabilistic Frame Induction. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 837–846, Atlanta, GA.
- Ciaramita, Massimiliano and Mark Johnson. 2003. Supersense tagging of unknown nouns in WordNet. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 168–175, Sapporo.
- Clark, Stephen. 2015. *Vector Space Models of Lexical Meaning*, 2nd edition. John Wiley & Sons, Inc.
- Cocos, Anne and Chris Callison-Burch. 2016. Clustering paraphrases by word sense. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1463–1472, San Diego, CA.
- Cotterell, Ryan, Adam Poliak, Benjamin Van Durme, and Jason Eisner. 2017. Explaining and generalizing skip-gram through exponential family principal component analysis. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 175–181, Valencia.
- Das, Dipanjan, Desai Chen, André F. T. Martins, Nathan Schneider, and Noah A. Smith. 2014. Frame-semantic parsing. *Computational Linguistics*, 40(1):9–56.
- Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, Miami Beach, FL.
- de Marneffe, Marie-Catherine, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 449–454, Genoa.
- de Saussure, Ferdinand. 1916. *Cours de linguistique générale*. Payot, Paris, France.
- Di Marco, Antonio and Roberto Navigli. 2013. Clustering and diversifying Web search results with graph-based word

- sense induction. *Computational Linguistics*, 39(3):709–754.
- Dikonov, Vyachelav G. 2013. Development of lexical basis for the Universal Dictionary of UNL Concepts. In *Computational Linguistics and Intellectual Technologies: Papers from the Annual International Conference "Dialogue,"* volume 12(19), pages 212–221, Moscow.
- van Dongen, Stijn. 2000. *Graph Clustering by Flow Simulation*. Ph.D. thesis, University of Utrecht.
- Dorow, Beate and Dominic Widdows. 2003. Discovering corpus-specific word senses. In *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics - Volume 2*, pages 79–82, Budapest.
- Dorow, Beate, Dominic Widdows, Katarina Ling, Jean-Pierre Eckmann, Danilo Sergi, and Elisha Moses. 2005. Using curvature and Markov clustering in graphs for lexical acquisition and word sense discrimination. In *Proceedings of the MEANING-2005 Workshop*, Trento.
- Dror, Rotem, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker’s guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, Melbourne.
- Egurnov, Dmitry, Dmitry Ignatov, and Engelbert Mephu Nguifo. 2017. Mining triclusters of similar values in triadic real-valued contexts. In *14th International Conference on Formal Concept Analysis - Supplementary Proceedings*, pages 31–47, Rennes.
- Erk, Katrin and Sebastian Padó. 2006. SHALMANESER — A toolchain for shallow semantic parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 527–532, Genoa.
- Everett, Martin and Stephen P. Borgatti. 2005. Ego network betweenness. *Social Networks*, 27(1):31–38.
- Evert, Stefan. 2005. *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. Ph.D. thesis, University of Stuttgart.
- Faralli, Stefano, Alexander Panchenko, Chris Biemann, and Simone Paolo Ponzetto. 2016. Linked disambiguated distributional semantic networks, In *The Semantic Web – ISWC 2016: 15th International Semantic Web Conference, Proceedings, Part II*, pages 56–64, Kobe.
- Faruqui, Manaal, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, Denver, CO.
- Fellbaum, Christiane. 1998. *WordNet: An Electronic Database*. MIT Press.
- Ferraresi, Adriano, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukWaC, a very large Web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4): Can We Beat Google?*, pages 47–54, Marrakech.
- Fillmore, Charles J. 1982. In Frame semantics, In *Linguistics in the Morning Calm*, Hanshin Publishing Co., pages 111–137, Seoul.
- Firth, John R. 1957. A synopsis of linguistic theory 1930–1955, In *Studies in Linguistic Analysis*, Blackwell, Oxford, UK, pages 1–32.
- Fleiss, Joseph L. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.
- Flekova, Lucie and Iryna Gurevych. 2016. Supersense embeddings: A unified model for supersense interpretation, prediction, and utilization. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2029–2041, Berlin.
- Fortunato, Santo. 2010. Community detection in graphs. *Physics Reports*, 486(3):75–174.
- Gildea, Daniel and Martin Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.
- Goldhahn, Dirk, Thomas Eckart, and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the Leipzig Corpora Collection: From 100 to 200 languages. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 759–765, Istanbul.
- Gonçalo Oliveira, Hugo and Paolo Gomes. 2014. ECO and Onto.PT: A flexible approach for creating a Portuguese wordnet automatically. *Language Resources and Evaluation*, 48(2):373–393.
- Gong, Zhiguo, Chan Wa Cheang, and Leong Hou U. 2005. Web query expansion by WordNet. In *Proceedings of the 16th International Conference on Database and Expert Systems Applications*, pages 166–175, Copenhagen.
- Graff, David and Christopher Cieri. 2003. English Gigaword. <https://catalog.1dc.upenn.edu/1dc2003t05>.
- Gurevych, Iryna, Judith Eckle-Kohler, Silvana Hartmann, Michael Matuschek, Christian M. Meyer, and Christian Wirth.

2012. UBY – A large-scale unified lexical-semantic resource based on LMF. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 580–590, Avignon.
- Hanks, Patrick and James Pustejovsky. 2005. A pattern dictionary for natural language processing. *Revue Française de linguistique appliquée*, 10(2):63–82.
- Hartigan, John A. and M. Anthony Wong. 1979. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108.
- Hartmann, Silvana, Judith Eckle-Kohler, and Iryna Gurevych. 2016. Generating Training Data for Semantic Role Labeling based on Label Transfer from Linked Lexical Resources. *Transactions of the Association for Computational Linguistics*, 4:197–213.
- Hearst, Marti A. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th Conference on Computational Linguistics - Volume 2*, pages 539–545, Nantes.
- Hope, David and Bill Keller. 2013a. MaxMax: A graph-based soft clustering algorithm applied to word sense induction. In *Computational Linguistics and Intelligent Text Processing: 14th International Conference, CICLing 2013*, pages 368–381, Samos.
- Hope, David and Bill Keller. 2013b. UoS: A graph-based system for graded word sense induction. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 689–694, Atlanta, GA.
- Horký, Vojtěch, Peter Libič, Antonín Steinhauser, and Petr Tůma. 2015. DOs and DON'Ts of conducting performance measurements in Java. In *Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering*, pages 337–340, Austin, TX.
- Hovy, Dirk, Chunliang Zhang, Eduard Hovy, and Anselmo Peñas. 2011. Unsupervised discovery of domain-specific knowledge from text. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1466–1475, Portland, OR.
- Hovy, Eduard, RobertoNavigli, and Simone Paolo Ponzetto. 2013. Collaboratively built semi-structured content and artificial intelligence: The story so far. *Artificial Intelligence*, 194:2–27.
- Huang, Eric H., Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, pages 873–882, Jeju Island.
- Ignatov, Dmitry I., Dmitry V. Gnatyshak, Sergei O. Kuznetsov, and Boris G. Mirkin. 2015. Triadic formal concept analysis and triclustering: searching for optimal patterns. *Machine Learning*, 101(1–3):271–302.
- Jauhar, Sujay Kumar and Eduard Hovy. 2017. Embedded semantic lexicon induction with joint global and local optimization. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM) (*SEM 2017)*, pages 209–219, Vancouver.
- Jurgens, David and Ioannis Klapaftis. 2013. SemEval-2013 Task 13: Word sense induction for graded and non-graded senses. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 290–299, Atlanta, GA.
- Kallmeyer, Laura, Behrang QasemiZadeh, and Jackie Chi Kit Cheung. 2018. Coarse lexical frame acquisition at the syntax–semantics interface using a latent-variable PCFG model. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 130–141, New Orleans, LA.
- Kawahara, Daisuke, Daniel W. Peterson, and Martha Palmer. 2014. A step-wise usage-based method for inducing polysemy-aware verb classes. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics Volume 1: Long Papers*, pages 1030–1040, Baltimore, MD.
- Kiselev, Yuri, Sergey V. Porshnev, and Mikhail Mukhin. 2015. Современное состояние электронных тезаурусов русского языка: качество, полнота и доступность [Current Status of Russian Electronic Thesauri: Quality, Completeness and Availability]. *Programmnaya Ingeneria*, 6:34–40. In Russian.
- Korhonen, Anna, Yuval Krymolowski, and Zviika Marx. 2003. Clustering polysemic subcategorization frame distributions semantically. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics - Volume 1*, pages 64–71, Sapporo.

- Kozareva, Zornitsa, Ellen Riloff, and Eduard Hovy. 2008. Semantic class learning from the Web with hyponym pattern linkage graphs. In *Proceedings of ACL-08: HLT*, pages 1048–1056, Columbus, OH.
- Krizhanovsky, Andrew A. and Alexander V. Smirnov. 2013. An approach to automated construction of a general-purpose lexical ontology based on Wiktionary. *Journal of Computer and Systems Sciences International*, 52(2):215–225.
- Kwok, Cody, Oren Etzioni, and Daniel S. Weld. 2001. Scaling question answering to the Web. *ACM Transactions on Information Systems*, 19(3):242–262.
- Lang, Joel and Mirella Lapata. 2010. Unsupervised Induction of Semantic Roles. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 939–947, Los Angeles, CA.
- Laparra, Egoitz and German Rigau. 2010. eXtended WordFrameNet. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, pages 1214–1219, Valletta.
- Lewis, Mike and Mark Steedman. 2013a. Combined distributional and logical semantics. *Transactions of the Association of Computational Linguistics*, 1:179–192.
- Lewis, Mike and Mark Steedman. 2013b. Unsupervised induction of cross-lingual semantic relations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 681–692, Seattle, WA.
- Li, Jiwei and Dan Jurafsky. 2015. Do multi-sense embeddings improve natural language understanding? In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1722–1732, Lisbon.
- Lin, Dekang. 1998. An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 296–304, Madison, WI.
- Lin, Dekang and Patrick Pantel. 2001. Induction of semantic classes from natural language text. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 317–322, San Francisco, CA.
- Loukachevitch, Natalia V. 2011. Тезаурусы в задачах информационного поиска [*Thesauri in Information Retrieval Tasks*]. Moscow University Press, Moscow. In Russian.
- Loukachevitch, Natalia V., German Lashevich, Anastasia A. Gerasimova, Vladimir V. Ivanov, and Boris V. Dobrov. 2016. Creating Russian WordNet by conversion. In *Computational Linguistics and Intellectual Technologies: Papers from the Annual Conference "Dialogue 2016,"* pages 405–415, Moscow.
- Manandhar, Suresh, Ioannis Klapaftis, Dmitriy Dligach, and Sameer Pradhan. 2010. SemEval-2010 Task 14: Word sense induction & disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 63–68, Uppsala.
- Materna, Jiří. 2012. In LDA-frames: An unsupervised approach to generating semantic frames, In *13th International Conference, Proceedings, Part I*, pages 376–387, New Delhi.
- Materna, Jiří. 2013. Parameter estimation for LDA-frames. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 482–486, Atlanta, GA.
- McIntosh, Tara and James R. Curran. 2009. Reducing semantic drift with bagging and distributional similarity. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 396–404, Suntec.
- McNemar, Quinn. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.
- Medelyan, Olena. 2007. Computing lexical chains with graph clustering. In *Proceedings of the 45th Annual Meeting of the ACL: Student Research Workshop*, pages 85–90, Prague.
- Meyer, Christian M., Margot Mieskes, Christian Stab, and Iryna Gurevych. 2014. DKPro agreement: An open-source Java library for measuring inter-rater agreement. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: System Demonstrations*, pages 105–109, Dublin.
- Mihalcea, Rada and Dragomir Radev. 2011. *Graph-based Natural Language Processing and Information Retrieval*. Cambridge University Press, Cambridge, UK.
- Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119, Las Vegas, NV.

- Mirkin, Boris. 1996. Clustering algorithms: A review, In *Mathematical Classification and Clustering*. Springer US, Boston, MA, pages 109–168.
- Modi, Ashutosh, Ivan Titov, and Alexandre Klementiev. 2012. Unsupervised induction of frame-semantic representations. In *Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure*, pages 1–7, Montréal.
- Montejo-Ráez, Arturo, Eugenio Martínez-Cámara, M. Teresa Martín-Valdivia, and L. Alfonso Ureña López. 2014. Ranked WordNet graph for sentiment polarity classification in Twitter. *Computer Speech & Language*, 28(1):93–107.
- Narayanan, Srinivasan, Collin Baker, Charles Fillmore, and Miriam Petruck. 2003. FrameNet meets the semantic Web: Lexical semantics for the Web. In *The Semantic Web - ISWC 2003: Second International Semantic Web Conference, Proceedings*. pages 771–787, Sanibel Island, FL.
- Navigli, Roberto and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.
- Neelakantan, Arvind, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1059–1069, Doha.
- Newman, Mark E. J. and Michelle Girvan. 2004. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113.
- Ng, Vincent. 2007. Semantic class induction and coreference resolution. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 536–543, Prague.
- Nieto Piña, Luis and Richard Johansson. 2015. A simple and efficient method to generate word sense representations. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 465–472, Hissar.
- Nivre, Joakim, Johan Hall, and Jens Nilsson. 2006. MaltParser: A data-driven parser-generator for dependency parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 2216–2219, Genoa.
- O'Connor, Brendan. 2013. Learning frames from text with an unsupervised latent variable model. Technical report, Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA.
- Padó, Sebastian and Mirella Lapata. 2009. Cross-lingual annotation projection of semantic roles. *Journal of Artificial Intelligence Research*, 36(1):307–340.
- Palla, Gergely, Imre Derenyi, Illes Farkas, and Tamás Vicsek. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814–818.
- Panchenko, Alexander, Stefano Faralli, Eugen Ruppert, Steffen Remus, Hubert Naets, Cedrick Fairon, Simone Paolo Ponzetto, and Chris Biemann. 2016a. TAXI at SemEval-2016 Task 13: A taxonomy induction method based on lexico-syntactic patterns, substrings and focused crawling. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, pages 1320–1327, San Diego, CA.
- Panchenko, Alexander, Eugen Ruppert, Stefano Faralli, Simone Paolo Ponzetto, and Chris Biemann. 2018a. Building a Web-scale dependency-parsed corpus from common crawl. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, pages 1816–1823, Miyazaki.
- Panchenko, Alexander, Johannes Simon, Martin Riedl, and Chris Biemann. 2016b. Noun sense induction and disambiguation using graph-based distributional semantics. In *Proceedings of the 13th Conference on Natural Language Processing*, pages 192–202, Bochum.
- Panchenko, Alexander, Dmitry Ustalov, Nikolay Arefyev, Denis Paperno, Natalia Konstantinova, Natalia Loukachevitch, and Chris Biemann. 2017. Human and machine judgements for Russian semantic relatedness. In *Analysis of Images, Social Networks and Texts: 5th International Conference, Revised Selected Papers*, pages 221–235, Yekaterinburg.
- Panchenko, Alexander, Dmitry Ustalov, Stefano Faralli, Simone Paolo Ponzetto, and Chris Biemann. 2018b. Improving hypernymy extraction with distributional semantic classes. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, pages 1541–1551, Miyazaki.
- Pang, Bo and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics*, pages 133–144, Barcelona, Spain.

- Linguistics Main Volume*, pages 271–278, Barcelona.
- Pantel, Patrick, Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu, and Vishnu Vyas. 2009. Web-scale distributional similarity and entity set expansion. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 938–947, Singapore.
- Pantel, Patrick and Dekang Lin. 2002. Discovering word senses from text. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 613–619, Edmonton.
- Pantel, Patrick and Deepak Ravichandran. 2004. Automatically labeling semantic classes. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 321–328, Boston, MA.
- Pavlick, Ellie, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 425–430, Beijing.
- Pedersen, Ted and Rebecca Bruce. 1997. Distinguishing word senses in untagged text. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 197–207, Providence, RI.
- Pelevina, Maria, Nikolay Arefiev, Chris Biemann, and Alexander Panchenko. 2016. Making sense of word embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 174–183, Berlin.
- Pennington, Jeffrey, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, Doha.
- Peters, Matthew, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, LA.
- Qadir, Ashequl, Pablo N. Mendes, Daniel Gruhl, and Neal Lewis. 2015. Semantic lexicon induction from Twitter with pattern relatedness and flexible term length. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, AAAI-15*, pages 2432–2439, Austin, TX.
- Reisinger, Joseph and Raymond J. Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117, Los Angeles, CA.
- Richardson, Ray, Alan F. Smeaton, and John Murphy. 1994. Using WordNet as a knowledge base for measuring semantic similarity between words. Working Paper CA-1294, School of Computer Applications, Dublin City University, Dublin, Ireland.
- Richter, Matthias, Uwe Quasthoff, Erla Hallsteinsdóttir, and Chris Biemann. 2006. Exploiting the Leipzig corpora collection. In *Proceedings of 5th Slovenian and 1st International Language Technologies Conference*, Ljubljana. http://nl.ijs.si/isjt06/proc/13_Richter.pdf
- Riedl, Martin. 2016. *Unsupervised Methods for Learning and Using Semantics of Natural Language*. Ph.D. thesis, Technische Universität Darmstadt.
- Riedl, Martin and Chris Biemann. 2017. There's no 'count or predict' but task-based selection for distributional models. In *Proceedings of the 12th International Conference on Computational Semantics — Short papers*, pages 264–272, Montpellier.
- Ritter, Alan, Mausam, and Oren Etzioni. 2010. A latent Dirichlet allocation method for selectional preferences. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 424–434, Uppsala.
- Rong, Xin, Zhe Chen, Qiaozhu Mei, and Eytan Adar. 2016. EgoSet: Exploiting word Ego-networks and user-generated ontology for multifaceted set expansion. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 645–654, San Francisco, CA.
- Ruppert, Eugen, Jonas Klesy, Martin Riedl, and Chris Biemann. 2015. Rule-based dependency parse collapsing and propagation for German and English. In *Proceedings of the International Conference of the German Society for Computational*

- Linguistics and Language Technology*, pages 58–66, Duisburg and Essen.
- Salton, Gerard, Andrew Wong, and Chungshu S. Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.
- Sarmiento, Luis, Valentin Jijkuon, Maarten de Rijke, and Eugenio Oliveira. 2007. “More like these”: Growing entity classes from seeds. In *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management*, pages 959–962, Lisbon.
- Schaeffer, Satu Elisa. 2007. Graph clustering. *Computer Science Review*, 1(1):27–64.
- Schenk, Niko and Christian Chiarcos. 2016. Unsupervised learning of prototypical fillers for implicit semantic role labeling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1473–1479, San Diego, CA.
- Schütze, Hinrich. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.
- Seabold, Skipper and Josef Perktold. 2010. Statsmodels: Econometric and statistical modeling with Python. In *Proceedings of the 9th Python in Science Conference*, pages 57–61, Austin, TX.
- Shen, Dan and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 12–21, Prague.
- Shen, Jiaming, Zequi Wu, Dongming Lei, Jingbo Shang, Xiang Ren, and Jiawei Han. 2017. SetExpan: Corpus-based set expansion via context feature selection and rank ensemble. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Proceedings, Part I*, pages 288–304, Skopje.
- Shi, Jianbo and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.
- Steyvers, Mark and Joshua B. Tenenbaum. 2005. The large-scale structure of semantic networks: Statistical analyses and a model of semantic growth. *Cognitive Science*, 29(1):41–78.
- Sundheim, Beth M. 1992. Overview of the fourth message understanding evaluation and conference. In *Proceedings of the 4th Conference on Message Understanding*, pages 3–21, McLean, VA.
- Talukdar, Partha Pratim, Joseph Reisinger, Marius Pasca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. Weakly-supervised acquisition of labeled class instances using graph random walks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 582–590, Honolulu, HI.
- Thelen, Michael and Ellen Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 214–221, Pennsylvania, PA.
- Thomason, Jesse and Raymond J. Mooney. 2017. Multi-modal word synset induction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 4116–4122, Melbourne.
- Tian, Fei, Hanjun Dai, Jiang Bian, Bin Gao, Rui Zhang, Enhong Chen, and Tie-Yan Liu. 2014. A probabilistic model for learning multi-prototype word embeddings. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 151–160, Dublin.
- Titov, Ivan and Alexandre Klementiev. 2011. A Bayesian model for unsupervised semantic parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1445–1455, Portland, OR.
- Titov, Ivan and Alexandre Klementiev. 2012. A Bayesian approach to unsupervised semantic role induction. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 12–22, Avignon.
- Tonelli, Sara and Daniele Pighin. 2009. New features for FrameNet - WordNet mapping. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 219–227, Boulder, CO.
- Turney, Peter D. and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.
- Ustalov, Dmitry, Mikhail Chernoskutov, Chris Biemann, and Alexander Panchenko. 2017. Fighting with the sparsity of the synonymy dictionaries for automatic synset induction. In *Analysis of Images, Social Networks and Texts: 6th International Conference, Revised Selected Papers*, pages 94–105, Moscow.

- Ustalov, Dmitry and Alexander Panchenko. 2017. A tool for effective extraction of synsets and semantic relations from BabelNet. In *Proceedings of the 2017 Siberian Symposium on Data Science and Engineering*, pages 10–13, Novosibirsk.
- Ustalov, Dmitry, Alexander Panchenko, and Chris Biemann. 2017. Watset: Automatic induction of synsets from a graph of synonyms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1579–1590, Vancouver.
- Ustalov, Dmitry, Alexander Panchenko, Andrei Kutuzov, Chris Biemann, and Simone Paolo Poncet. 2018. Unsupervised semantic frame induction using triclustering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 55–62, Melbourne.
- Véronis, Jean. 2004. HyperLex: Lexical cartography for information retrieval. *Computer Speech & Language*, 18(3):223–252.
- Wang, Richard C. and William W. Cohen. 2008. Iterative set expansion of named entities using the Web. In *2008 Eighth IEEE International Conference on Data Mining*, pages 1091–1096, Pisa.
- Welch, Bernard Lewis. 1947. The generalization of ‘Student’s’ problem when several different population variances are involved. *Biometrika*, 34(1–2):28–35.
- Widdows, Dominic and Beate Dorow. 2002. A graph model for unsupervised lexical acquisition. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, pages 1–7, Taipei.
- Xie, Boyi, Rebecca J. Passonneau, Leon Wu, and Germán G. Creamer. 2013. Semantic frames to predict stock price movement. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 873–883, Sofia.
- Zesch, Torsten, Christof Müller, and Iryna Gurevych. 2008. Extracting lexical semantic knowledge from Wikipedia and Wiktionary. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, pages 1646–1652, Marrakech.
- Zhao, Lizhuang and Mohammed J. Zaki. 2005. TRICLUSTER: An effective algorithm for mining coherent clusters in 3d microarray Data. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, pages 694–705, New York, NY.
- Zhou, Guangyou, Yang Liu, Fang Liu, Daojian Zeng, and Jun Zhao. 2013. Improving question retrieval in community question answering using world knowledge. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, pages 2239–2245, Beijing.
- Zipf, George K. 1949. *Human Behavior and the Principle of Least Effort*. Addison-Wesley, Menlo Park, CA.

A.7 Paper “Linked Disambiguated Distributional Semantic Networks”

S. Faralli, [A. Panchenko](#), C. Biemann, and S. P. Ponzetto, “**Linked Disambiguated Distributional Semantic Networks**,” in *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part II* (P. Groth, E. Simperl, A. J. G. Gray, M. Sabou, M. Krötzsch, F. Lécué, F. Flöck, and Y. Gil, eds.), vol. 9982 of *Lecture Notes in Computer Science*, pp. 56–64, 2016

Permission to copy: the article is available in the public domain on the website at the following address: https://doi.org/10.1007/978-3-319-46547-0__7

Linked Disambiguated Distributional Semantic Networks

Stefano Faralli^{1(✉)}, Alexander Panchenko²,
Chris Biemann², and Simone P. Ponzetto¹

¹ Data and Web Science Group, University of Mannheim, Mannheim, Germany

{stefano,simone}@informatik.uni-mannheim.de

² Language Technology Group, TU Darmstadt, Darmstadt, Germany

{panchenko,biem}@lt.informatik.tu-darmstadt.de

Abstract. We present a new hybrid lexical knowledge base that combines the contextual information of distributional models with the conciseness and precision of manually constructed lexical networks. The computation of our count-based distributional model includes the induction of word senses for single-word and multi-word terms, the disambiguation of word similarity lists, taxonomic relations extracted by patterns and context clues for disambiguation in context. In contrast to dense vector representations, our resource is human readable and interpretable, and thus can be easily embedded within the Semantic Web ecosystem.

Resource Type: Lexical Knowledge Base

Permanent URL: <https://madata.bib.uni-mannheim.de/id/eprint/171>

1 Introduction

Recent years have witnessed an impressive amount of work on the automatic construction of wide-coverage knowledge resources from Wikipedia [3, 13] and the Web [7]. Complementary to this, a plethora of works in Natural Language Processing (NLP) has recently focused on combining knowledge bases with distributional information from text. These include approaches that modify Word2Vec [15] to learn sense embeddings [5], methods to enrich WordNet with embeddings for synsets and lexemes [21], acquire continuous word representations by combining random walks over knowledge bases and neural language models [11], or produce joint lexical and semantic vectors for sense representation from text and knowledge bases [4].

In this paper, we follow this line of research and take it one step forward by producing a hybrid knowledge resource, which combines symbolic and statistical meaning representations while (i) staying purely on the lexical-symbolic level, (ii) explicitly distinguishing word senses, and (iii) being human readable. Far from being technicalities, such properties are crucial to be able to embed a resource of this kind into the Semantic Web ecosystem, where human-readable

distributional representations are explicitly linked to URIified semantic resources. To this end, we develop a methodology to automatically induce distributionally-based semantic representations from large amounts of text, and link them to a reference knowledge base. This results in a new knowledge resource that we refer to as a *hybrid aligned resource*.

2 Building a Hybrid Aligned Resource

Our resource is built in three main phases (Fig. 1):

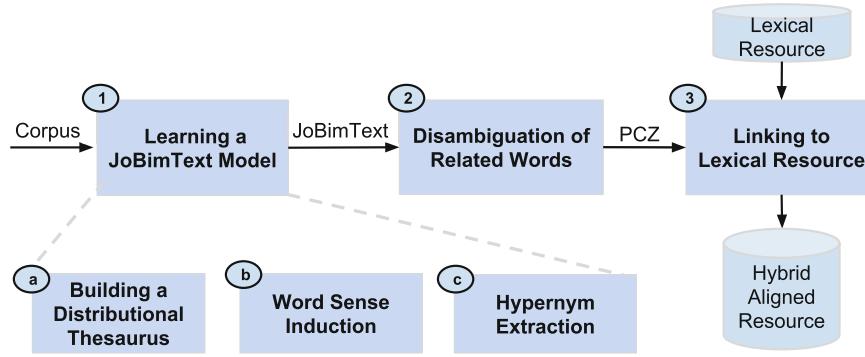


Fig. 1. Overview of our method for constructing a hybrid aligned resource.

- (1) **Learning a JoBimText model:** initially, we automatically create a sense inventory from a large text collection using the pipeline of the JoBimText project [2, 22]¹. The resulting structure contains disambiguated proto-concepts (i.e. senses), their similar and related terms, as well as aggregated contextual clues per proto-concept (Table 1a). This is a distributionally-based conceptualization with some degree of taxonomic information only. Hence the two subsequent phases, which, together with the final resource, represent the novel contribution of this paper.
- (2) **Disambiguation of related terms:** we fully disambiguate all lexical information associated with a proto-concept (i.e. similar terms and hypernyms) based on the partial disambiguation from step 1). The result is a protoconceptualization (PCZ). In contrast to a term-based distributional thesaurus (DT), a PCZ consists of sense-disambiguated entries, i.e. all terms have a sense identifier (Table 1b).
- (3) **Linking to a lexical resource:** we align the PCZ with an existing lexical resource (LR). That is, we create a mapping between the two sense inventories and then combine them into a new extended sense inventory, our *hybrid aligned resource*.

¹ <http://www.jobimtext.org>.

2.1 Learning a JoBimText Model

Following [2], we apply a holing operation where each observation in the text is split into a term and its context. The 1000 most significant contexts per term, as determined by the LMI significance measure [8], serve as a representation for the term, and term similarity is defined as the number of common contexts. This procedure induces a weighted similarity graph over terms, also known as Distributional Thesaurus (DT), where each entry of the DT consists of the most similar 200 terms for a given term.

In DTs, entries of polysemous terms t are mixed, i.e. they contain similar terms stemming from several senses respectively usages of the term. Since terms that belong to the same sense are more similar to each other than to terms belonging to a different sense, we can employ graph clustering to partition the open neighbourhood of t in the DT (i.e., terms similar to t and their similarities, without t) to arrive at sense representations for t , characterized by a list of similar terms. We achieve this by applying Chinese Whispers [1] on the ego-network of the term t , as defined by its similar terms as nodes.

Table 1. Examples of entries for “mouse” and “keyboard” from the *news-p1.6* dataset before and after the semantic closure. Trailing numbers indicate sense identifiers.

entry	similar terms	hypernyms	context clues
mouse>NN:0	rat>NN, rodent>NN, monkey>NN, ...	animal>NN, species>NN, ...	rat>NN:conj_and, white-footed:JJ:amod, ...
mouse>NN:1	keyboard>NN, computer>NN, printer>NN, ...	device>NN, equipment>NN, ...	click>NN:-prep_of, click>NN:-nn, ...
keyboard>NN:0	piano>NN, synthesizer>NN, organ>NN, ...	instrument>NN, device>NN, ...	play:VB:-dobj, electric:JJ:amod, ...
keyboard>NN:1	keypad>NN, mouse>NN, screen>NN, ...	device>NN, technology>NN, ...	computer>NN:nn, qwerty:JJ:amod, ...

(a) JoBimText model entries

entry	similar terms	hypernyms	context clues
mouse>NN:0	rat>NN:0, rodent>NN:0, monkey>NN:0, ...	animal>NN:0, species>NN:1, ...	rat>NN:conj_and, white-footed:JJ:amod, ...
mouse>NN:1	keyboard>NN:1, computer>NN:0, printer>NN:0, ...	device>NN:1, equipment>NN:3, ...	click>NN:-prep_of, click>NN:-nn, ...
keyboard>NN:0	piano>NN:1, synthesizer>NN:2, organ>NN:0, ...	instrument>NN:2, device>NN:3, ...	play:VB:-dobj, electric:JJ:amod, ...
keyboard>NN:1	keypad>NN:0, mouse>NN:1, screen>NN:1, ...	device>NN:1, technology>NN:0, ...	computer>NN:nn, qwerty:JJ:amod, ...

(b) Proto-conceptualization entries

Further, we run Hearst patterns [12] over the corpus to extract IS-A (hypernym) relations between terms. We add these hypernyms to senses by aggregating IS-A relations over the list of similar terms for the given sense into a weighted list of hypernyms. Additionally, we aggregate the significant contexts of similar terms per sense to arrive at weighted aggregated context clues. The resulting structure is called the JoBimText model [22] of the corpus. A JoBimText entry consists of a distributionally-induced word sense, a ranked list of similar terms for this sense, a list of superordinate terms and a list of aggregated context clues (note that only unstructured text is required). Table 1(a) shows some JoBimText entries for the polysemous terms “mouse” and “keyboard”.

2.2 Disambiguation of Related Terms

While JoBimText models contain sense distinctions, they are not fully disambiguated: the list of similar and hypernyms terms of each sense does not carry sense information. In our example (Table 1a) the sense of “mouse:NN” for the entry “keyboard:NN:1” could either be the “animal” or the “electronic device” one. Consequently, we next apply a semantic closure procedure to arrive at a PCZ in which *all* terms get assigned a unique, best-fitting sense identifier (Table 1b).

At its heart, our method assigns each target word w to disambiguate – namely, a similar and superordinate term from each sense of the JoBimText model – the sense \hat{s} whose context (i.e., the list of similar or superordinate terms) has the maximal similarity with the target word’s context (i.e., the other words in the target word’s list of similar or superordinate items) – we use cosine as similarity metric:

$$\hat{s} = \operatorname{argmax}_{s \in Senses_{JoBim}(w)} \cos(ctx(w), ctx(s)). \quad (1)$$

This way we are able to link, for instance, *keyboard:NN* in the list of similar terms for *mouse:NN:1* to its ‘device’ sense (*keyboard:NN:1*), since *mouse:NN:1* and *keyboard:NN:1* share a large amount of terms from the IT domain.

The structure of a PCZ resembles that of a lexical semantic resource: each term has a list of proto-concepts, and proto-concepts are linked via relations, such as similarity and taxonomic links. Sense distinctions and distributions are dependent on the underlying corpus, which causes the PCZ to naturally adapt to the domain of the corpus. A large difference to manually created resources, however, is the availability of aggregated context clues that allow to disambiguate polysemous terms in text with respect to their sense distinctions. Table 1(b) shows example proto-concepts for the terms “mouse:NN” and “keyboard:NN”, taken from our *news-p1.6* PCZ (see Sect. 3.1).

2.3 Linking to a Lexical Resource

We next link each sense in our proto-conceptualization (PCZ) to the most suitable sense (if any) of a Lexical Resource (LR, see Fig. 1 step 3). Our method takes as input:

1. a PCZ $T = \{(j_i, R_{j_i}, H_{j_i})\}$ where j_i is a sense identifier (i.e. *mouse:NN:1*), R_{j_i} the set of its semantically related senses (i.e. $R_{j_i} = \{\text{keyboard:NN:1}, \text{computer:NN:0}, \dots\}$) and H_{j_i} the set of its hypernym senses (i.e. $H_{j_i} = \{\text{equipment:NN:3}, \dots\}$);
2. a LR W : we experiment with: WordNet [10], a lexical database for English and BabelNet [16], a very large multilingual “encyclopaedic dictionary”;
3. a threshold th over the similarity between pairs of concepts and a number m of iterations as stopping criterion;

and outputs a mapping M , which consists of a set of pairs of the kind $(source, target)$ where $source \in T.senses$ is a sense of the input PCZ T and

$target \in W.senses \cup source$ is the most suitable sense of W or $source$ when no such sense is available. At its heart, the mapping algorithm compares the senses across resources with the following similarity function:

$$sim(j, c, M) = \frac{|T.BoW(j, M, W) \cap W.BoW(c)|}{|T.BoW(j, M, W)|}, \text{ where:} \quad (2)$$

1. $T.BoW(j, M, W)$ is the set of words containing all the terms extracted from related/hypernym senses of j and all the terms extracted from the related/hypernym (i.e., already linked in M) synsets in W . For each synset we use all synonyms and content words of the gloss.
2. $W.BoW(c)$ contains the synonyms and the gloss content words for the synset c and all the related synsets of c .

A new link pair (j, c) is then added to M if the similarity score between j and c is greater than or equal to a threshold th . Finally, all unlinked j of T , i.e. proto-concepts that have no corresponding LR sense, are added to the mapping M . We follow the guidelines from McCrae et al. [14] and create an RDF representation to share the mapping between our PCZs and lexical knowledge graphs (i.e., WordNet and BabelNet) in the Linked Open Data Cloud [6].

3 Experiments

3.1 Datasets for the Extraction of the Proto-Conceptualizations (PCZs)

We experiment with two different large corpora, namely a 100 million sentence news corpus (*news*) from Gigaword [17] and LCC [19], and with a 35 million sentence Wikipedia corpus (*wiki*) and different parametrizations of the sense induction algorithm to obtain five proto-conceptualizations (PCZ) with different average sense granularities. Further, we use the method described in [20] to compute a dataset that includes automatically extracted multiword terms. In Table 2, we present figures for our five datasets. For each dataset, Columns 3, 4 and 5 count the overall number of words, including monosemous words and polysemous ones, respectively. For each PCZ we report the cardinality (Column 6), the average polysemy (Column 7) and the maximum polysemy (Column 8). Finally, we report the overall and the average number of related senses and hypernyms (Column 9–12).

3.2 Experiment 1: Disambiguation of the Distributional Thesaurus Entries

Experimental setting. In order to disambiguate a related or superordinate term t in a word sense entry s in the JoBimText model, we compare the related words of s with the related words of each of the senses t_s for the target term t . Similarly, we evaluate the quality of the disambiguation of the JoBimText models by judging the compatibility of the similar words for s and t_s .

Table 2. Structural analysis of our five proto-conceptualizations (PCZs).

dataset	words				senses	polysemy		rel. senses		hypernyms	
	n	#	monosemous	polysemous	#	avg.	max	#	avg.	#	avg.
news-p1.6	200	207 k	137 k	69 k	332k	1.6	18	234 k	63.9	15 k	6.9
news-p2.3	50	200 k	99 k	101 k	461 k	2.3	17	298 k	44.3	15 k	5.8
wiki-p1.8	200	206 k	120 k	86 k	368 k	1.8	15	300 k	59.3	15 k	4.4
wiki-p6.0	30	258 k	44 k	213 k	1.5 M	6.0	36	811 k	16.9	52 k	1.7
wiki-mw-p1.6	200	465 k	288 k	176 k	765 k	1.6	13	662 k	46.6	30 k	3.2

For instance, the similar term *mouse:NN*, in the JoBimText model entry for keyboard>NN:1, namely “keypad>NN, *mouse:NN*, screen>NN, ...” is compatible with the related words “keyboard>NN, computer>NN, printer>NN ...” (i.e., those of sense mouse>NN:1) and is not compatible with the related words “cat>NN, rodent>NN, monkey>NN, ...” of mouse>NN:0 (see Table 1).

Our experimental setting is based on three steps: (1) we manually select 17 highly ambiguous *target words*; (2) we collect 19,774 disambiguated entries of the *wiki-p1.6* JoBimText model where the target words appear and randomly sample 15 % of these entries to make annotation feasible; (3) we manually judge entries in the sample on whether the related words of the target word fits the sense assigned or not². Finally, we compute performance by means of standard accuracy – i.e., the proportion of cases in which the similar or hypernym terms from the JoBimText model are correctly disambiguated.

Results and discussion. Our method achieves an accuracy of 0.84 across all parts of speech, including accuracy of 0.94 for nouns, 0.85 for proper nouns, 0.76 for verbs, and 0.63 for adjectives. Different results across parts of speech are due to the different quality of the respective DT clusters. This is because this first experiment also indirectly measures the quality of the senses from the JoBimText model: indeed, a match between two word sense entries is only possible if both of them are interpretable.

To better understand the amount of spurious items in our sense clusters, we performed an additional manual evaluation where, for a sample of 100 randomly sampled noun PCZ items, we counted the ratio between wrong (e.g., *rat* for the computer sense of *mouse*) and correct (*keyboard*, *computer*, etc.) related concepts that were found within the PCZs. We obtained a macro average of 0.0495 and a micro average of 0.0385 wrong related concepts within the PCZs. Moreover, 83 % of the above sample has no unrelated concepts, and only 2 % has only one unrelated concept with a macro average ratio between the wrong and correct related PCZ of 0.067. This indicates that, overall, the amount of spurious concepts within clusters is indeed small, thus providing a high-quality context for an accurate disambiguation of noun DT clusters.

² The target words and annotations can be found at <https://goo.gl/jjdhl4>.

3.3 Experiment 2: Linking to Lexical Knowledge Bases

Experimental setting. Next, we evaluate the performance of our linking component (Sect. 2.3). For this, we choose two lexical-semantic networks: WordNet [10], which has a high coverage on English common nouns, verbs and adjectives, and BabelNet [16], which also includes a large amount of proper nouns as well as senses gathered from multiple other sources, including Wikipedia.

We follow standard practices (e.g., [16]) and create five evaluation test sets, one for each dataset from Sect. 3.1, by randomly selecting a subset of 300 proto-concepts for each dataset, and manually establishing a mapping from these senses to WordNet and BabelNet concepts (proto-concepts that cannot be mapped are labeled as such in the gold standard). The quality and correctness of the mapping is estimated as accuracy on the ground-truth judgments, namely the amount of true mapping decisions among the total number of (potentially, empty) mappings in the gold standard. We also evaluate our mapping by quantifying Coverage (the percentage of senses of the knowledge base sense inventory covered by the mapping M) and ExtraCoverage (the ratio of concepts in M not linked to the knowledge base sense inventory over the total number of knowledge base senses). The latter is a measure of novelty to quantify the amount of senses discovered in T and not represented by the knowledge base: it indicates the amount of ‘added’ knowledge we gain with our resource based on the amount of proto-concepts that cannot be mapped and are thus included as novel senses.

Table 3. Results on linking to lexical knowledge bases: number of linked proto-concepts (\hat{J}), Coverage, ExtraCoverage and Accuracy for our five datasets.

Dataset	WordNet				BabelNet			
	\hat{J} senses	Cov.	ExtraCov.	Accuracy	\hat{J} senses	Cov.	ExtraCov.	Accuracy
news-p1.6	88 k	34.5 %	206.0 %	86.9 %	164 k	1.3 %	2.9 %	81.8 %
news-p2.3	145 k	38.2 %	267.0 %	93.3 %	236 k	1.4 %	3.9 %	85.1 %
wiki-p1.8	91 k	35.9 %	234.7 %	94.8 %	232 k	1.9 %	2.4 %	86.4 %
wiki-p6.0	400 k	49.9 %	919.9 %	93.5 %	737 k	2.8 %	1.3 %	82.2 %
wiki-mw-p1.6	81 k	30.7 %	581.2 %	95.3 %	589 k	4.7 %	1.8 %	83.8 %

Results and discussion. In Table 3 we present the results using the optimal parameter values (i.e. $th = 0.0$ and $m = 5$)³. For all datasets the number of linked senses, Coverage and ExtraCoverage are directly proportional to the number of entries in the dataset – i.e., the finer the concept granularity, as given by a lower sense clustering n parameter, the lower the number of mapped senses, Coverage and ExtraCoverage.

³ To find optimal value for m , we prototyped our approach on a dev set consisting of a random sample of 300 proto-concepts, and studied the curves for the number of linked proto-concepts to WordNet resp. BabelNet. The th value was then selected as to maximize the accuracy.

In general, we report rather low coverage figures: the coverage in WordNet is always lower than 50% (30% in one setting) and in BabelNet is in all settings lower than 5%. Low coverage is due to different levels of granularities between the source and target resource. Our target knowledge bases, in fact, have very fine-grained sense inventories. For instance, BabelNet lists 17 senses of the word “python” including two (arguably obscure ones) referring to particular roller coasters. In contrast, word senses induced from text corpora tend to be coarse and corpus-specific. Consequently, the low coverage comes from the fact that we connect a coarse and a fine-grained sense inventory – cf. also previous work [9] showing comparable proportions between coverage and extra-coverage of automatically acquired knowledge (i.e., glosses) from corpora.

Finally, our results indicate differences between the order of magnitude of the Coverage and ExtraCoverage when linking to WordNet and BabelNet. This high difference depends on the cardinality of the two sense inventories, where BabelNet has millions of senses while WordNet more than a hundred of thousands – many of them not covered in our corpora. Please note that an ExtraCoverage of about 3% in BabelNet corresponds to about 300k novel senses. Overall, we take our results to be promising in that, despite the relative simplicity of our approach (i.e., almost parameter-free unsupervised linking), we are able to reach high accuracy figures in the range of around 87%–95% for WordNet and accuracies above 80% for BabelNet.

4 Conclusions

We presented an automatically-constructed hybrid aligned resource that combines distributional semantic representations with lexical knowledge graphs. To the best of our knowledge, we are the first to present such a large-scale, fully URIified hybrid aligned resource with high alignment quality. As future work, we will explore ways to couple focused crawling [18] with domain-specific PCZs to extend our resource to many domains. Moreover, we aim at using our linguistically-grounded hybrid resource to provide generalizations beyond concepts, such as, for instance, hybrid symbolic and distributional representations of actions and events.

Acknowledgments. We acknowledge the support of the Deutsche Forschungsgemeinschaft (DFG) under the JOIN-T project.

References

1. Biemann, C.: Chinese Whispers: an efficient graph clustering algorithm and its application to natural language processing problems. In: Proceedings of the TextGraphs, pp. 73–80 (2006)
2. Biemann, C., Riedl, M.: Text: now in 2D! a framework for lexical expansion with contextual similarity. J. Lang. Model. **1**(1), 55–95 (2013)
3. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - a crystallization point for the web of data. JWS **7**(3), 154–165 (2009)

4. Camacho-Collados, J., Pilehvar, M.T., Navigli, R.: NASARI: a novel approach to a semantically-aware representation of items. In: Proceedings of the NAACL-HLT, pp. 567–577 (2015)
5. Chen, X., Liu, Z., Sun, M.: A unified model for word sense representation and disambiguation. In: Proceedings of the EMNLP, pp. 1025–1035 (2014)
6. Chiarcos, C., Hellmann, S., Nordhoff, S.: Linking linguistic resources: examples from the open linguistics working group. In: Chiarcos, C., Nordhoff, S., Hellmann, S. (eds.) *Linked Data in Linguistics - Representing and Connecting Language Data and Language Metadata*, pp. 201–216. Springer, Heidelberg (2012)
7. Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., Zhang, W.: Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In: Proceedings of the KDD, pp. 601–610 (2014)
8. Evert, S.: The Statistics of Word Cooccurrences: Word Pairs and Collocations. Ph.D. thesis, Institut für maschinelle Sprachverarbeitung, University of Stuttgart (2005)
9. Faralli, S., Navigli, R.: Growing multi-domain glossaries from a few seeds using probabilistic topic models. In: Proceedings of the EMNLP, pp. 170–181 (2013)
10. Fellbaum, C. (ed.): *WordNet: An Electronic Database*. MIT Press, Cambridge (1998)
11. Goikoetxea, J., Soroa, A., Agirre, E.: Random walks and neural network language models on knowledge bases. In: Proceedings of the NAACL HLT, pp. 1434–1439 (2015)
12. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the COLING, pp. 539–545 (1992)
13. Hoffart, J., Suchanek, F.M., Berberich, K., Weikum, G.: YAGO2: a spatially and temporally enhanced knowledge base from Wikipedia. *ArtInt*, pp. 28–61 (2013)
14. McCrae, J.P., Fellbaum, C., Cimiano, P.: Publishing and linking WordNet using lemon and RDF. In: Proceedings of the 3rd Workshop on Linked Data in Linguistics (2014)
15. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of the NIPS, pp. 3111–3119 (2013)
16. Navigli, R., Ponzetto, S.P.: BabelNet: the automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *ArtInt.* **193**, 217–250 (2012)
17. Parker, R., Graff, D., Kong, J., Chen, K., Maeda, K.: English Gigaword, 5th edn. Linguistic Data Consortium, Philadelphia (2011)
18. Remus, S., Biemann, C.: Domain-specific corpus expansion with focused webcrawling. In: Proceedings of the LREC (2016)
19. Richter, M., Quasthoff, U., Hallsteinsdóttir, E., Biemann, C.: Exploiting the Leipzig corpora collection. In: Proceedings of the IS-LTC (2006)
20. Riedl, M., Biemann, C.: A single word is not enough: ranking multiword expressions using distributional semantics. In: Proceedings of the EMNLP, pp. 2430–2440 (2015)
21. Rothe, S., Schütze, H.: AutoExtend: extending word embeddings to embeddings for synsets and lexemes. In: Proceedings of the ACL, pp. 1793–1803 (2015)
22. Ruppert, E., Kaufmann, M., Riedl, M., Biemann, C.: JoBimViz: a web-based visualization for graph-based distributional semantic models. In: Proceedings of the ACL-IJCNLP System Demonstrations, pp. 103–108 (2015)

A.8 Article “A framework for enriching lexical semantic resources with distributional semantics”

C. Biemann, S. Faralli, A. Panchenko, and S. P. Ponzetto, “**A framework for enriching lexical semantic resources with distributional semantics**,” *Nat. Lang. Eng.*, vol. 24, no. 2, pp. 265–312, 2018

Permission to copy: the article is available in the public domain on the website at the following address: <https://doi.org/10.1017/S135132491700047X>

A framework for enriching lexical semantic resources with distributional semantics

CHRIS BIEMANN¹, STEFANO FARALLI²,
ALEXANDER PANCHENKO¹ and
SIMONE PAOLO PONZETTO²

¹*Language Technology Group, Department of Informatics, Faculty of Mathematics,
Informatics, and Natural Sciences, Universität Hamburg, Germany*
e-mails: {biemann, panchenko}@informatik.uni-hamburg.de

²*Data and Web Science Group, School of Business Informatics and Mathematics,
Universität Mannheim, Germany*
e-mails: {stefano, simone}@informatik.uni-mannheim.de

(Received 28 July 2016; revised 12 December 2017; accepted 15 December 2017)

Abstract

We present an approach to combining distributional semantic representations induced from text corpora with manually constructed lexical semantic networks. While both kinds of semantic resources are available with high lexical coverage, our aligned resource combines the domain specificity and availability of contextual information from distributional models with the conciseness and high quality of manually crafted lexical networks. We start with a distributional representation of induced senses of vocabulary terms, which are accompanied with rich context information given by related lexical items. We then automatically disambiguate such representations to obtain a full-fledged proto-conceptualization, i.e. a typed graph of induced word senses. In a final step, this proto-conceptualization is aligned to a lexical ontology, resulting in a hybrid aligned resource. Moreover, unmapped induced senses are associated with a semantic type in order to connect them to the core resource. Manual evaluations against ground-truth judgments for different stages of our method as well as an extrinsic evaluation on a knowledge-based Word Sense Disambiguation benchmark all indicate the high quality of the new hybrid resource. Additionally, we show the benefits of enriching top-down lexical knowledge resources with bottom-up distributional information from text for addressing high-end knowledge acquisition tasks such as cleaning hypernym graphs and learning taxonomies from scratch.

1 Introduction

Automatic enrichment of semantic resources is an important problem (Biemann 2005; Jurgens and Pilehvar 2016) as it attempts to alleviate the extremely costly process of *manual* lexical resource (LR) construction. Distributional semantics (Turney and Pantel 2010; Baroni, Dinu and Kruszewski 2014; Clark 2015) provides an alternative *automatic* meaning representation framework that has been shown to benefit many text-understanding applications.

Recent years have witnessed an impressive amount of work on combining the symbolic semantic information available in manually constructed LRs with distributional information, where words are usually represented as dense numerical vectors, a.k.a. embeddings. Examples of such approaches include methods that modify the Skip-gram model (Mikolov *et al.* 2013) to learn sense embeddings (Chen, Liu and Sun 2014) based on the sense inventory of WordNet, methods that learn embeddings of synsets as given in an LR (Rothe and Schütze 2015) or methods that acquire word vectors by applying random walks over LRs to learn a neural language model (Goikoetxea, Soroa and Agirre 2015). Besides, alternative approaches like NASARI (Camacho-Collados, Pilehvar and Navigli 2015b) and MUFFIN (Camacho-Collados, Pilehvar and Navigli 2015a) looked at ways to produce joint lexical and semantic vectors for a common representation of word senses in text and in LRs. Retrofitting approaches, e.g. Faruqui *et al.* (2015), efficiently ‘consume’ LRs as input in order to improve the quality of word embeddings, but do not add anything to the resource itself. Other approaches, such as AutoExtend (Rothe and Schütze 2015), NASARI and MUFFIN, learn vector representations for existing synsets that can be added to the resource, however are not able to add missing senses discovered from text.

In these contributions, the benefits of such hybrid knowledge sources for tasks in computational semantics such as semantic similarity and Word Sense Disambiguation (WSD) (Navigli 2009) have been extensively demonstrated. However, none of the existing approaches, to date, have been designed to use distributional information for the enrichment of lexical semantic resources, i.e. adding new symbolic entries.

In this article, we consequently set out to filling this gap by developing a framework for enriching lexical semantic resources with distributional semantic models. The goal of such framework is the creation of a resource that brings together the ‘best of both worlds’, namely the *precision* and *interpretability* from the lexicographic tradition that describes senses and semantic relations manually, and the *versatility* of data-driven, corpus-based approaches that derive senses automatically.

An LR enriched with additional knowledge induced from text can boost the performance of natural language understanding tasks like WSD or Entity Linking (Mihalcea and Csomai 2007; Rospocher *et al.* 2016), where it is crucial to have a comprehensive list of word senses as well as the means to assign the correct of many possible senses for a given word in context.

Consider, for instance, the following sentence:

Regulator of calmodulin signaling (RCS) knockout mice display anxiety-like behavior and motivational deficits.¹

No synset for ‘RCS’ can be found in either WordNet (Fellbaum 1998) or BabelNet (Navigli and Ponzetto 2012a), yet it is distributionally related to other biomedical concepts and thus can help to disambiguate the ambiguous term *mice* to its ‘animal’ meaning, as opposed to the ‘computer peripheral device’.

Our approach yields a hybrid resource that combines symbolic and statistical meaning representations while (i) staying purely on the lexical-symbolic level,

¹ <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3622044>

(ii) explicitly distinguishing word senses and (iii) being human readable. These properties are crucial to be able to embed such a resource in an explicit semantic data space such as, for instance, the Linguistic Linked Open Data ecosystem (Chiarcos, Hellmann and Nordhoff 2012). According to Norvig (2016), the semantic web and natural language understanding are placed at the heart of current efforts to understand the web on a large scale.

We take the current line of research on hybrid semantic representations one step forward by presenting a methodology for inducing distributionally based sense representations from text, and for linking them to a reference LR. Central to our method is a novel sense-based distributional representation that we call proto-conceptualization (PCZ). A PCZ is a repository of word senses induced from text, where each sense is represented with related senses, hypernymous senses and aggregated clues for contexts in which the respective sense occurs in text. Besides, to further improve interpretability, each sense has an associated image. We build a bridge between the PCZ and lexical semantic networks by establishing a mapping between these two kinds of resources.² This results in a new knowledge resource that we refer to as *hybrid aligned resource* (HAR); here, senses induced from text are aligned to a set of synsets from a reference LR, whereas induced senses that cannot be matched are included as additional synsets. By linking our distributional representations to a repository of symbolically encoded knowledge, we are able to complement wide-coverage statistical meaning representations with explicit relational knowledge as well as to extend the coverage of the reference LR based on the senses induced from a text collection. The main contributions of this article are listed as follows:

- **A framework for enriching lexical semantic resources:** We present a methodology for combining information from distributional semantic models with manually constructed lexical semantic resources.
- **A hybrid lexical semantic resource:** We apply our framework to produce a novel hybrid resource obtained by linking disambiguated distributional lexical semantic networks to WordNet and BabelNet.
- **Applications of the hybrid resource:** Besides *intrinsic* evaluations of our approach, we test the utility of our resource *extrinsically* on the tasks of WSD and taxonomy induction, demonstrating the benefits of combining distributional and symbolic lexical semantic knowledge.

The remainder of this article is organized as follows: we first review related work in Section 2 and provide an overview of our framework to build a HAR from distributional semantic vectors and a reference knowledge graph in Section 3. Next, we provide details on our methodology to construct PCZs and to link them to an LR in Sections 4 and 5, respectively. In Section 6, we benchmark the quality of our resource in different evaluation settings, including an intrinsic and an extrinsic evaluation on the task of knowledge-based WSD using a dataset from a SemEval task. Section 7 provides two application-based evaluations that demonstrate how

² We use WordNet and BabelNet; however, our approach can be used with similar resources, e.g. those listed at <http://globalwordnet.org/wordnets-in-the-world>.

our hybrid resource can be used for taxonomy induction. We conclude with final remarks and future directions in Section 8.

2 Related work

2.1 Automatic construction of lexical semantic resources

In the past decade, large efforts have been undertaken to research the automatic acquisition of machine-readable knowledge on a large scale by mining large repositories of textual data (Banko *et al.* 2007; Carlson *et al.* 2010; Fader, Soderland and Etzioni 2011; Faruqui and Kumar 2015). At this, collaboratively constructed resources have been exploited, used either in isolation (Bizer *et al.* 2009; Ponzetto and Strube 2011; Nastase and Strube 2013) or complemented with manually assembled knowledge sources (Suchanek, Kasneci and Weikum 2008; Gurevych *et al.* 2012; Navigli and Ponzetto 2012a; Hoffart *et al.* 2013). As a result of this, recent years have seen a remarkable renaissance of knowledge-rich approaches for many different artificial intelligence tasks (Hovy, Navigli and Ponzetto 2013). Knowledge contained within these very large knowledge repositories, however, has major limitations in that these resources typically do not contain any linguistically grounded probabilistic representation of concepts, instances and their attributes – namely, the bridge between wide-coverage conceptual knowledge and its instantiation within natural language texts. While there are large-scale LRs derived from large corpora such as ProBase (Wu *et al.* 2012), these are usually not sense aware but conflate the notions of term and concept. With this work, we provide a framework that aims at augmenting any of these wide-coverage knowledge sources with distributional semantic information, thus extending them with text-based contextual information.

Another line of research has looked at the problem of Knowledge Base Completion (Nickel *et al.* 2016). Many approaches to Knowledge Base Completion focus on exploiting other KBs (Wang *et al.* 2012; Bryl and Bizer 2014) for acquiring additional knowledge, or rely on text corpora – either based on distant supervision (Snow, Jurafsky and Ng 2006; Mintz *et al.* 2009; Aprasio, Giuliano and Lavelli 2013) or by rephrasing KB relations as queries to a search engine (West *et al.* 2014) that returns results from the web as corpus. Alternative methods primarily rely on existing information in the KB itself (Bordes *et al.* 2011; Jenatton *et al.* 2012; Socher *et al.* 2013; Klein, Ponzetto and Glavaš 2017) to simultaneously learn continuous representations of KB concepts and KB relations by exploiting the KB structure as the ground truth for supervision, inferring additional relations from existing ones. Lexical semantic resources and text are synergistic sources, as shown by complementary work from Faruqui *et al.* (2015), who improve the quality of semantic vectors based on lexicon-derived relational information.

Here, we follow this intuition of combining structured knowledge resources with distributional semantic information from text, but focus instead on providing hybrid semantic representations for KB concepts and entities, as opposed to the classification task of Knowledge Base Completion that aims at predicting additional semantic relations between known entities.

2.2 Combination of distributional semantics with lexical resources

Several prior approaches combined distributional information extracted from text with information available in LRs like e.g. WordNet. This includes a model (Yu and Dredze 2014) to learn word embeddings based on lexical relations of words from WordNet and PPDB (Ganitkevitch, Van Durme and Callison-Burch 2013). The objective function of this model combines that of the skip-gram model (Mikolov *et al.* 2013) with a term that takes into account lexical relations of target words. In work aimed at retrofitting word vectors (Faruqui *et al.* 2015), a related approach was proposed that performs post-processing of word embeddings on the basis of lexical relations from LRs. Finally, Pham, Lazaridou and Baroni (2015) also aim at improving word vector representations by using lexical relations from WordNet, targeting similar representations of synonyms and dissimilar representations of antonyms. While all these three approaches show excellent performance on word relatedness evaluations, they do not model word senses – in contrast to other work aimed instead at learning sense embeddings using the word sense inventory of WordNet (Jauhar, Dyer and Hovy 2015).

A parallel line of research has recently focused on learning unified statistical and symbolic semantic representations. Approaches aimed at providing unified semantic representations from distributional information and LRs have accordingly received an increasing level of attention (Chen *et al.* 2014; Goikoetxea, Soroa and Agirre 2015; Rothe and Schütze 2015; Camacho-Collados *et al.* 2015b; Nieto Piña and Johansson 2016), *inter alia* (cf. also our introductory discussion in Section 1) and hybrid meaning representations have been shown to benefit challenging semantic tasks such as WSD and semantic similarity at word level and text level.

All these diverse contributions indicate the benefits of hybrid knowledge sources for learning word and sense representations; here, we further elaborate along this line of research and develop a new hybrid resource that combines information from the knowledge graph with distributional sense representations that are human readable and easy to interpret, in contrast to dense vector representations, a.k.a. word embeddings like GloVe (Pennington, Socher and Manning 2014) or word2vec (Mikolov *et al.* 2013). As a result of this, we are able to provide, to the best of our knowledge, the first hybrid knowledge resource that is fully integrated and embedded within the semantic web ecosystem provided by the Linguistic Linked Open Data cloud (Chiarcos, Hellmann and Nordhoff 2012). Note that this complementary to recent efforts aimed at linking natural language expressions in text with semantic relations found within LOD knowledge graphs (Krause *et al.* 2015), in that we focus instead on combining explicit semantic information with statistical, distributional semantic representations of concepts and entities into an augmented resource.

3 Enriching lexical semantic resources with distributional semantics

The construction of our HAR builds upon methods used to link various manually constructed LRs to construct BabelNet (Ponzetto andNavigli 2010) and UBY (Gurevych *et al.* 2012), among others. In our method, however, linking is performed between two networks that are structurally similar, but have been

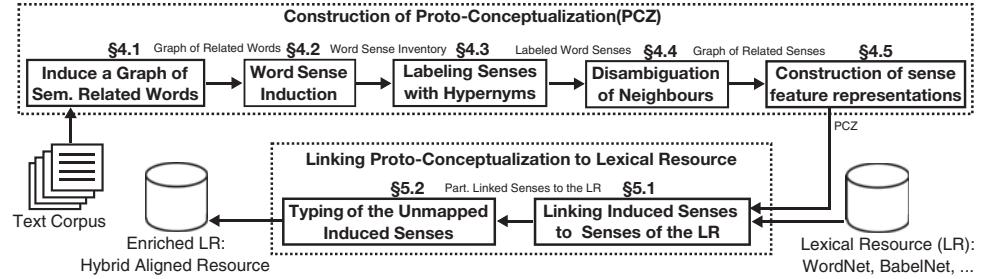


Fig. 1. Overview of the proposed framework for enriching lexical resources: a distributional semantic model is used to construct a disambiguated distributional lexical semantic network (a proto-conceptualization, PCZ), which is subsequently linked to the lexical resource.

Table 1. Examples of entries of the PCZ resource for words *mouse* and *keyboard* after disambiguation of their related terms and hypernyms (Section 4.4)

Word sense	Related senses	Hypernyms	Context clues
mouse:0	rat:0, rodent:0, monkey:0, ...	animal:0, species:1, ...	rat:conj_and, white-footed:amod, ...
mouse:1	keyboard:1, printer:0, computer:0 ...	device:1, equipment:3, ...	click:-prep_of, click:-nn, ...
keyboard:0	piano:1, synthesizer:2, organ:0 ...	instrument:2, device:3, ...	play:-dobj, electric:amod, ..
keyboard:1	keypad:0, mouse:1, screen:1 ...	device:1, technology:0 ...	computer:nn, qwerty:amod ...

Context clues are represented as typed dependency relations to context words in the input corpus, e.g. *keyboard:0* appears as direct object of ‘to play’. Trailing numbers indicate automatically induced sense identifiers.

constructed in two completely different ways: one resource is built using an unsupervised bottom-up approach from text corpora, while the second is constructed in a top-down manner using manual labor, e.g. codified knowledge from human experts such as lexicographers (WordNet). In particular, the method consists of following two major phases, as illustrated in Figure 1:

- (1) **Construction of a PCZ from text.** Initially, a symbolic disambiguated distributional lexical semantic network, called a PCZ, is induced from a text corpus. To this end, we first create a sense inventory from a large text collection using graph-based word sense induction as provided by the JoBimText project (Biemann and Riedl 2013). The resulting structure contains induced word senses, their yet undisambiguated related terms, as well as context clues per term. First, we obtain sense representations by aggregating context clues over sense clusters. Second, we disambiguate related terms and hypernyms to produce a fully disambiguated resource where all terms have a sense identifier. Hence, the PCZ is a repository of corpus-induced word senses, where each sense is associated with a list of related senses, hypernymous senses, as well as aggregated contextual clues (Table 1).
- (2) **Linking a PCZ to an LR.** Next, we align the PCZ with an existing LR. In our work, we make use of lexical semantic resources such as WordNet and BabelNet featuring large vocabularies of lexical units with explicit meaning representations as well as semantic relations between these. In this phase, we

create a mapping between the two sense inventories from the PCZ and the LR, and combine them into a new extended sense inventory, our HAR. Finally, to obtain a complete unified resource, we link the ‘orphan’ PCZ senses for which no corresponding sense could be found by inferring their type (i.e. their most specific generalization) in the LR.

In the following sections, we present each stage of our approach in detail.

4 Construction of a proto-conceptualization

Our method for PCZ construction consists of the four steps illustrated in the upper half of Figure 1, inducing a graph of semantically related words (Section 4.1), word sense induction (Section 4.2), labeling of clusters with hypernyms and images (Section 4.3) and disambiguation of related words and hypernyms with respect to the induced sense inventory (Section 4.4). Further, we describe an additional property of our PCZs, namely the availability of corpus-derived context clues (Section 4.5), as well as alternative ways to construct PCZs on the basis of dense vector representations (Section 4.6).

The PCZ resource with a pipeline as outlined in Figure 1 consists of word senses induced from a corpus. For each word sense, similar and superordinate terms are disambiguated with respect to the induced sense inventory: the structure of a PCZ resembles that of a lexical semantic resource. Sense distinctions and distributions depend on the training corpus, which causes the resource to adapt to its domain. In contrast to manually created resources, each sense also contains context clues that allow disambiguating polysemous terms in context. Table 1 shows example senses for the terms *mouse* and *keyboard*. Note that PCZs may contain many entries for the same word, e.g. *mouse* has two senses, the ‘animal’ and the ‘computer device’, respectively. The context clues are not disambiguated, since they are meant for directly matching (undisambiguated) textual context. PCZs can be interpreted by humans at two levels, as exemplified in Figure 2.

- (1) The **word sense inventory** is interpretable due to the use of the hypernyms, images and related senses.
- (2) The **sense feature representation** is interpretable due to the use of the sparse context clues relevant to the sense.

Note that while in our experiments we rely on a count-based sparse distributional model, the PCZ is a symbolic structure that can be also constructed using alternative distributional models, e.g. word and sense embeddings (cf. Section 4.6).

4.1 Inducing a graph of semantically related words

The goal of this first stage is to build a graph of semantically related words, with edges such as (*mouse*, *keyboard*, 0.78), i.e. a distributional thesaurus (DT) (Lin 1998). To induce such graph in an unsupervised way, we rely on a count-based approach to distributional semantics based on the *JoBimText* framework (Biemann and Riedl 2013). Each word is represented by a bag of syntactic dependencies

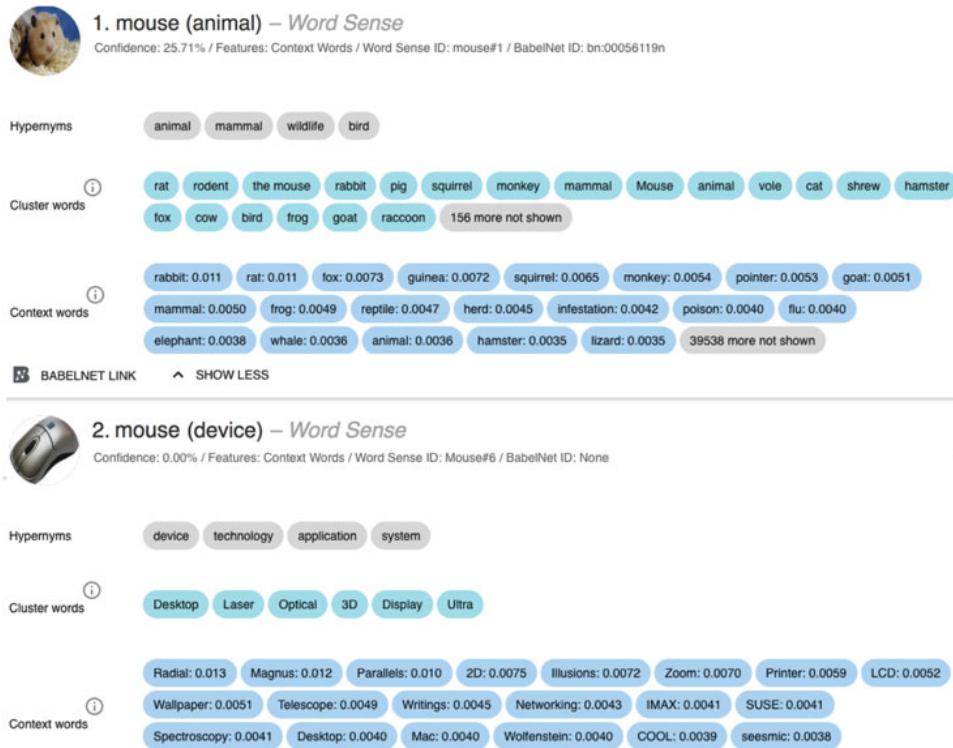


Fig. 2. (Colour online) Word sense representations of the word *mouse* induced from text generated using the online demo at <http://jobimtext.org/wsd>. The sense labels (device and animal) are obtained automatically based on cluster labeling with hypernyms. The images associated with the senses are retrieved with a search engine using the queries: *mouse device* and *mouse animal*. Note the ‘BabelNet Link’ button, leading to the sense in BabelNet linked to the induced sense with the algorithm described in Section 5.

such as `conj_and(rat, •)` or `prep_of(click,•)` extracted from the Stanford Dependencies (De Marneffe, MacCartney and Manning 2006) obtained with the PCFG model of the Stanford parser (Klein and Manning 2003).

Features of each word are weighted and ranked using the Local Mutual Information (LMI) metric (Evert 2005). Subsequently, these word representations are pruned keeping 1,000 most salient features per word and 1,000 most salient words per feature. The pruning reduces computational complexity and noise (Riedl 2016). Finally, word similarities are computed as the number of common features for two words. This is, again, followed by a pruning step in which for every word, only the 200 most similar terms are kept. The resulting graph of word similarities is browsable online (Ruppert et al. 2015).³

There are many possible ways to compute a graph of semantically similar words, including count-based approaches, such as Lin (1998), Curran (2002) or prediction-based approaches, such as word2vec (Mikolov et al. 2013), GloVe (Pennington et al.

³ Word and sense representations used in our experiments can be inspected by selecting the ‘Stanford (English)’ model in the JoBimViz demo at <http://jobimtext.org/jobimviz/>.

Table 2. Comparison of state-of-the-art count- and prediction-based methods to distributional semantics on the basis of the average of the averaged similarity scores between each term in the DT and its top-10 most similar terms using the WordNet path similarity measure (higher means better) averaged over 1,000 high- and low-frequency words

Method	High	Low
Lin's similarity (Lin 1998)	0.2872	0.2291
<i>t</i> -test (Curran 2002)	0.2589	0.2067
Skip-gram (Mikolov <i>et al.</i> 2013)	0.2548	0.2068
Skip-gram with dependency features (Levy and Goldberg 2014)	0.2632	0.1992
LMI with trigram features (Riedl and Biemann 2013)	0.2621	0.2003
LMI with dependency features (Riedl and Biemann 2013)	0.2933	0.2337

In this article, we use 'LMI with dependency features' as the similarity function.

2014) and word2vecf (Levy and Goldberg 2014). Here, we opt for a count-based approach to distributional semantics based on Local Mutual Information based on two considerations, namely their higher quality of similarity scores and their interpretability.

A thorough experimental comparison of different approaches to computing distributional semantic similarity to build a DT is presented by Riedl (2016, Section 5.7.4) using the WordNet taxonomy as a gold standard. In this evaluation, different DTs are compared by computing, for each term, the average similarity between the term itself and its k most similar terms (based on the DT) using the WordNet path-based similarity measure (Pedersen, Patwardhan and Michelizzi 2004). The overall similarity of the DT with the ground-truth taxonomy (e.g. WordNet) is then given by the average similarity score across all terms. Using this evaluation framework, Riedl is able to compare a wide range of different approaches for the construction of a weighted similarity graph, including state-of-the-art approaches based on sparse vector representations (Lin 1998; Curran 2002), as well as dense representations based on word2vec (Mikolov *et al.* 2013) and word2vecf, which makes use of dependency-based features (Levy and Goldberg 2014). In his experiment, all methods were trained on the same corpus, and all dependency-based models, including the Skip-gram approach trained with the word2vecf tool (Levy and Goldberg 2014), used the same feature representations.

We report some of the results from Riedl's experiments in Table 2. In this experiment, 1,000 infrequent and 1,000 frequent nouns proposed by Weeds, Weir and McCarthy (2004) were used. Dependency-based models (all models except the Skip-gram) used syntactic features extracted using the Stanford Parser. In addition to this dependency-based model, we report results of the same model based on trigram features, where a context is formed by the concatenation of the two adjacent words. All models were trained on the 105 million sentences of newspaper data described in Section 6.1. Further details of the experiment, e.g. parameters of the models, are available in Riedl (2016, Section 5.7.4).

The performance figures indicate that the method we use here yields the overall best performance in terms of semantic similarity compared to other count-based or

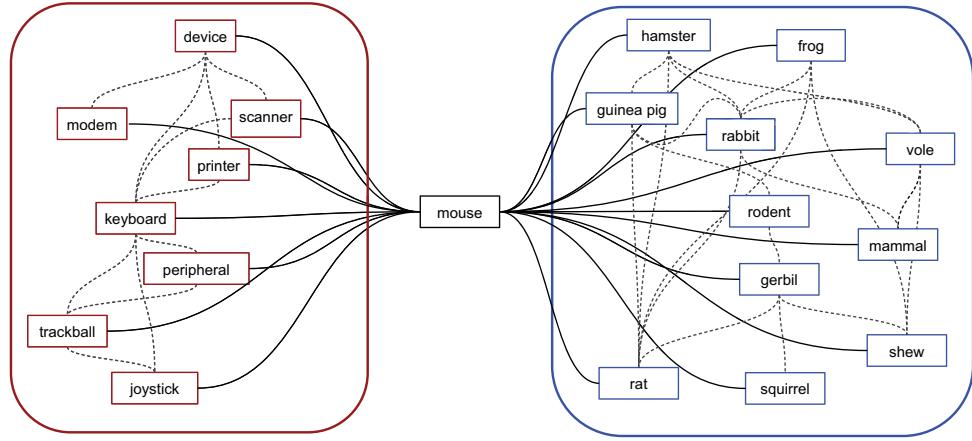


Fig. 3. (Colour online) Example of graph-based word sense induction for the word *mouse*: the two circles denote two induced word senses, as found by analysis of the ego graph of *mouse*.

word-embedding approaches (including both word2vec and word2vecf). Besides, the results generally indicate the advantage of using dependency-based context representations over the bag-of-words representations. This is in line with prior studies on semantic similarity (Padó and Lapata 2007; Van de Cruys 2010; Panchenko and Morozova 2012). For this reason, we use syntactic features in our experiments but would like to stress that the overall framework also allows simpler context representations, giving rise to its application to resource-poor languages.

The second reason for using the Local Mutual Information approach to compute a graph of semantically related words is the fact that the resulting word representations are human interpretable, since words are represented by sparse features – as opposed to dense features such as those found within word embeddings. Besides being a value on its own, this feature enables a straightforward implementation of word sense disambiguation methods on the basis of the learned representations (Panchenko *et al.* 2017a, 2017c).

4.2 Word sense induction

In the next stage, we induce a sense inventory on top of the DT by clustering ego-networks of similar words. In our case, an inventory represents senses by a word cluster, such as $\{\text{rat}, \text{rodent}, \text{monkey}, \dots\}$ for the ‘animal’ sense of the word *mouse*.

Sense induction is conducted one word t at the time on the DT. First, we retrieve nodes of the ego-network G of t being the N most similar words of t according to the DT. Figure 3 presents a sample ego network of related words.⁴ Note that the target word t itself is excluded during clustering. Second, we connect each node in G to its n most similar words according to DT. The n parameter regulates the granularity of the induced sense inventory: we experiment with $n \in \{200, 100, 50\}$ and $N = 200$.

⁴ See the Serelex.org system for further visualizations of ego networks of semantically related words (Panchenko *et al.* 2013).

Finally, the ego network is clustered with Chinese Whispers (Biemann 2006), a non-parametric algorithm that discovers the number of clusters (word senses, in our case) automatically. The algorithm is iterative and proceeds in a bottom-up fashion. Initially, all nodes have distinct cluster identifiers. At each step, a node obtains the cluster identifier from the *dominant* cluster in its direct neighborhood, which is the cluster with the highest sum of edge weights to the current node.

The choice of Chinese Whispers among other algorithms, such as HyperLex (Véronis 2004) or Markov Cluster Algorithm (Van Dongen 2008), was motivated by the absence of meta-parameters, its state-of-the-art performance on Word Sense Induction tasks (Di Marco and Navigli 2013), as well as its efficiency (time-linear in the number of edges), see Cecchini, Riedl and Biemann (2017) for a comparative evaluation.

4.3 Labeling induced senses with hypernyms and images

At the third stage, each sense cluster is automatically labeled to characterize it in more detail and to improve its interpretability. First, we extract hypernyms from the input corpus. Here, we rely on the Hearst (1992) patterns, yet the approach we use can benefit also from more advanced methods for extraction of hypernyms, e.g. HypeNet (Shwartz, Goldberg and Dagan 2016) or the Dual Tensor Model (Glavaš and Ponzetto 2017). Note that despite their simplicity, Hearst patterns still are a strong baseline, used for applications like, for instance, taxonomy induction (Panchenko *et al.* 2016; Bordea, Lefever and Buitelaar 2016).

Second, we rank the quality of a hypernym h to act as generalization for the meaning expressed by cluster c on the basis of the product of two scores, namely *hypernym relevance* and *coverage*:

$$\text{relevance}(c, h) = \sum_{w \in c} \text{rel}(t, w) \cdot \text{freq}(w, h)$$

$$\text{coverage}(c, h) = \sum_{w \in c} \min(\text{freq}(w, h), 1)$$

where $\text{rel}(t, w)$ is the relatedness of the cluster word w to the target word t (the ambiguous word in an ego network, cf. Figure 3) and $\text{freq}(w, h)$ is the frequency of the hypernymy relation (w, h) as extracted via patterns. Thus, a highly ranked hypernym h needs to be observed frequently in a hypernym pattern, but also needs to be confirmed by several cluster words. This stage results in a ranked list of labels that specify the word sense, which we add to the PCZ. The highest scoring hypernym is further used in the title of the word sense, e.g. *mouse (device)* or *mouse (animal)*.

Finally, to further improve the interpretability of the induced senses, we add images to our sense clusters as follows. Previous work (Faralli and Navigli 2012) showed that web search engines can be used to bootstrap sense-related information. Consequently, we assign an image to each word in the cluster querying the Bing image search API⁵ using the query composed of the target word and its highest

⁵ <https://azure.microsoft.com/en-us/services/cognitive-services/search>

Algorithm 1 Disambiguation of related terms and hypernyms

Require: WSI , a word sense inventory in the form of a set of tuples ($word, sense_id, cluster, isas$), where $cluster$ and $isas$ have no sense identifiers.

Ensure: PCZ , a proto-conceptualization in the form of a set of tuples ($word, sense_id, cluster', isas'$), where $cluster'$ and $isas'$ are disambiguated with respect to sense inventory of the WSI .

```

1:  $PCZ = \emptyset$ 
2: for all  $(word, sense\_id, cluster, isas) \in WSI$  do
3:    $cluster' = \text{DISAMBIGUATE}(cluster, word, WSI)$ 
4:    $isas' = \text{DISAMBIGUATE}(isas, word, WSI)$ 
5:    $PCZ = PCZ \cup (word, sense\_id, cluster', isas')$ 

function  $\text{DISAMBIGUATE}(cluster, cword, WSI)$ 
6:    $cluster' = \emptyset$ 
7:    $context = cluster \cup (cword, 1.0)$ 
8:   for all  $word, sim \in cluster$  do
9:      $sense\_id = -1, max\_sim = 0$ 
10:    for all  $(dword, dsense\_id, dcluster, disas) \in \text{GETSENSES}(word, WSI)$  do
11:      if  $sim(context, dcluster) > max\_sim$  then
12:         $sense\_id = dsense\_id$ 
13:         $max\_sim = sim$ 
14:     $cluster' = cluster' \cup (word, sim, sense\_id)$ 
15:   return  $cluster'$ 
```

scoring hypernym, e.g. mouse device. The first image result of this query is selected to represent the induced word sense. This step is optional in our pipeline, and is primarily aimed at improving the user interaction with the word sense inventory.

The resulting sense representation is illustrated in Figure 2 for two induced senses of the word *mouse*. Providing to the user hypernyms, images, list of related senses and the list of the most salient context clues ensures interpretability of each sense. Note that all these elements are obtained without manual intervention, see Panchenko *et al.* (2017b) for more details.

4.4 Disambiguation of related terms and hypernyms

Next, we disambiguate the lexical graphs induced in the previous step. Each word in the induced inventory has one or more senses; however, the list of related words and hypernyms of each induced sense does not carry sense information yet. In our example (Table 1), the sense of *mouse* for the entry *keyboard:1* could have either referred to the ‘animal’ or the ‘electronic device’. Consequently, we apply a semantic closure procedure to arrive at a resource in which all terms get assigned a unique, best-fitting sense identifier. Our method assigns each disambiguation target word w – namely, a similar or superordinate term from each sense of the induced word sense inventory – the sense \hat{s} whose context (i.e. the list of similar or superordinate terms) has the maximal similarity with the target word’s context (i.e. the other words in the target word’s list of similar or superordinate items). We use the cosine similarity between context vectors to find the most appropriate sense \hat{s} matching the ‘context’ of an ambiguous word $cluster$ one of its ‘definitions’ $WSI(w').cluster$:

$$\hat{s} = \underset{(w', _, cluster, _) \in WSI(w)}{\operatorname{argmax}} \cos(WSI(w').cluster, cluster). \quad (1)$$

This way we are able to link, for instance, `keyboard` in the list of similar terms for `mouse:1` to its ‘device’ sense (`keyboard:1`), since `mouse:1` and `keyboard:1` share a large amount of terms from the information technology domain. This simple, local approach is scalable (cf. the complexity analysis at the end of this section) and it performs well, as we show later in the evaluation.

Algorithm 1 presents our method to compute the semantic closure. The input is a JoBimText model as a set of tuples $(word, sense_id, cluster, isas)$, where *cluster* is a list of similar terms in the format $(word_i, sim_i)$ with sim_i being the similarity value between *word* and $word_i$, and *isas* is a list of hypernym terms in the same format. The algorithm outputs a PCZ in the form of a set of tuples $(word, sense_id, cluster', isas')$, where *cluster'* is a list of disambiguated similar terms and *isas'* is a list of disambiguated hypernym terms both in the format $(word_i, sim_i, sense_id_i)$. The algorithm starts by creating an empty PCZ structure *PCZ*. For each entry of an input JoBimText model, we disambiguate related words (*cluster*) and hypernym terms (*isas*) with the function *DISAMBIGUATE* (lines 3–4). This function retrieves for each *word* in a *cluster* the set of its senses with the *GETSENSES* function. Next, we calculate similarity between the *cluster* of the *word* and the cluster of the candidate sense (denoted as *dcluster*). The *word*_{*i*} obtains the *sense_id* of the candidate sense that maximizes this similarity (lines 8–13).

Our disambiguation approach is a rather straightforward algorithm based on similarity computations. Despite its simplicity, we are able to achieve a disambiguation accuracy in the high ninety per cent range for noun word senses, while at the same time having a time-linear complexity in the number of word senses, as we will show in the evaluation below (Section 6.2). We can assume, in fact, that the function *GETSENSES* has a run-time complexity of $O(1)$ and the function *cos* (Equation (1)) has complexity $O(m)$, where m is the average number of neighbors of each word sense. Then, the run-time complexity of the algorithm is $O(n * m^2 * k)$, where n is the overall number of induced word senses, and k is the average polysemy of a word. Since k is small and m is bound by the maximum number of neighbors (200 in our case), the amortized run time is linear in the vocabulary size. This makes our approach highly scalable: in recent experiments, we have been accordingly able to apply our method at web scale on the CommonCrawl,⁶ the largest existing public repository of web content.

4.5 Construction of sense feature representations

Finally, we calculate feature representations for each sense in the induced inventory – that is, grammatical dependency features that are meant to provide an aggregated representation of the contexts in which a word sense occurs.

We assume that a word sense is a composition of cluster words that represent the sense and accordingly define a sense vector as a function of word vectors representing cluster items. Let W be a set of all words in the training corpus and let $S_i = \{w_1, \dots, w_n\} \subseteq W$ be a sense cluster obtained in a previous step. Consider

⁶ <https://commoncrawl.org>

Table 3. *Sense inventories derived from the Wikipedia corpus via a sparse count-based (JoBimText) and dense predict-based (Skip-gram) distributional models*

	Sparse vectors (JoBimText)	Dense vectors (word2vec)
Mouse (animal)	Rat, rodent, monkey, pig, animal, human, rabbit, cow	Rat, hamster, hedgehog, mole, monkey, kangaroo, skunk
Mouse (device)	Keyboard, computer, printer, joystick, stylus, modem	Cursor, keyboard, AltGr, chording, D-pad, button, trackball

a function $\text{vec}_w : W \rightarrow \mathbb{R}^m$ that maps words to their vectors and a function $\gamma_i : W \rightarrow \mathbb{R}$ that maps cluster words to their weight in the cluster S_i . The sense vector representation (the context clues) is then a weighted average of word vectors:

$$S_i = \frac{\sum_{k=1}^n \gamma_i(w_k) \text{vec}_w(w_k)}{\sum_{k=1}^n \gamma_i(w_k)}. \quad (2)$$

Table 1 (column 4) provides an example of such feature representations. While the averaged word vectors are ambiguous and can contain features related to various senses, features with high weights tend to belong to the target sense as the secondary senses of the averaged words vectors rarely match semantically, hence the aggregation amplifies the correct sense.

This concludes the description of steps we use to construct PCZs from text corpora.

4.6 Inducing PCZs with dense vector representations

In this section, we briefly describe alternative routes to the construction of a PCZ from text in an unsupervised way. In the remainder of this article, we will rely on the results of the approach described above. The goal of this section is to show that our overall framework is agnostic to the type of underlying distributional semantic model. In this section, we consider three approaches to generating a PCZ using word or sense embeddings.

Option 1: Inducing PCZs using word embeddings with explicit disambiguation. As illustrated in Figure 1, the first stage of our approach involves the computation of a graph of semantically similar words. Above, the graph was induced using a count-based model, however, any of the models listed in Table 2 can be used to generate such a graph of ambiguous words including models based on dense vector representations, such as the Skip-gram model. In this strategy, one would need to generate top nearest neighbors of word on the basis of cosine similarity between word embeddings. Table 3 shows an excerpt of nearest neighbors generated using the JoBimText and word2vec toolkits. The obtained word graphs can be subsequently used to induce word senses using the graph-based approach described in Section 4.2. The obtained clusters can be labeled using a database of hypernyms exactly in the same way as for the models based on the count-based JoBimText framework

Table 4. A Skip-gram-based PCZ model by Pelevina *et al.* (2016): Neighbors of the word **mouse** and the induced senses

Vector	Nearest neighbors
Mouse	Rat, keyboard, hamster, hedgehog, monkey, kangaroo, cursor, button
Mouse:0	Rat:0, hamster:0, hedgehog:1, mole:0, monkey:0, kangaroo:0, skunk:0
Mouse:1	Cursor:0, keyboard:1, AltGr:0, chording:1, D-pad:0, button:0

The neighbors of the initial vector belong to both senses, while those of sense vectors are sense specific.

(cf. Section 4.3). All further steps of the workflow presented in Figure 1 remain the same.

The main difference between the approach described above and the methods based on dense representations of words is the representation of the context clues of PCZ (cf. Table 1). In the case of an underlying sparse count-based representation, context clues remain human readable and interpretable, whereas in case of dense representations, context clues are represented by a dense vector embedding, and it is not straightforward to aggregate context clues over sense clusters.

Option 2: Inducing PCZs using word embeddings without explicit disambiguation. The first two stages of this approach are the same compared to the previous strategy. Namely, first one needs to generate a graph of ambiguous semantically related words (Section 4.1) and then to run ego-network clustering to induce word senses (Section 4.2). However, instead of explicit disambiguation of nearest neighbors (Section 4.4), the third stage could obtain vector sense representations by averaging the word embeddings of sense clusters (Section 4.5). Finally, disambiguated nearest neighbors can be obtained by calculating nearest neighbors of each sense vector in the space of word sense embeddings. This step is equivalent to the computation of a DT (Section 4.1); however, it directly yields disambiguated nearest neighbors (cf. Table 4). Note, however that, disambiguation of hypernyms using Algorithm 1 is still required when using this approach.

This approach was explored in our previous work (Pelevina *et al.* 2016), where we showed that words sense embeddings obtained in this way can be successfully used for unsupervised WSD, yielding results comparable to the state of the art.

Option 3: Inducing PCZ using word sense embeddings. Finally, a PCZ can be also induced using sparse (Reisinger and Mooney 2010) and dense (Neelakantan *et al.* 2014; Li and Jurafsky 2015; Bartunov *et al.* 2016) multi-prototype vector space models (the latter are also known as word sense embeddings). These models directly induce sense vectors from a text corpus, not requiring the word sense induction step of our method (Section 4.2). Instead of ego-network-based sense induction, these methods rely on some form of context clustering, maintaining several vector representations for each word type during training. To construct a PCZ using such models within our framework, we need to compute a list of nearest neighbors (Section 4.1), label the obtained sense clusters with hypernyms (Section 4.3) and

Algorithm 2 Linking induced senses to senses of a lexical resource

Require: $T = \{(j_i, R_{j_i}, H_{j_i})\}$, W , th , m
Ensure: $M = (\text{source}, \text{target})$

- 1: $M = \emptyset$
- 2: **for all** $(j_i, R_{j_i}, H_{j_i}) \in T.\text{monosemousSenses}$ **do**
- 3: $C(j_i) = W.\text{getSenses}(j_i.\text{lemma}, j_i.\text{POS})$
- 4: **if** $|C(j_i)| == 1$, let $C(j_i) = \{c_0\}$ **then**
- 5: **if** $\text{sim}(j_i, c_0, \emptyset) \geq th$ **then**
- 6: $M = M \cup \{(j_i, c_0)\}$
- 7: **for** $\text{step} = 1$; $\text{step} \leq m$; $\text{step} = \text{step} + 1$ **do**
- 8: $M_{\text{step}} = \emptyset$
- 9: **for all** $(j_i, R_{j_i}, H_{j_i}) \in T.\text{senses}/M.\text{senses}$ **do**
- 10: $C(j_i) = W.\text{getSenses}(j_i.\text{lemma}, j_i.\text{POS})$
- 11: **for all** $c_k \in C(j_i)$ **do**
- 12: $\text{rank}(c_k) = \text{sim}(j_i, c_k, M)$
- 13: **if** $\text{rank}(c_k)$ has a single top value for c_t **then**
- 14: **if** $\text{rank}(c_t) \geq th$ **then**
- 15: $M_{\text{step}} = M_{\text{step}} \cup \{(j_i, c_t)\}$
- 16: $M = M \cup M_{\text{step}}$
- 17: **for all** $(j_i, R_{j_i}, H_{j_i}) \in T.\text{senses}/M.\text{senses}$ **do**
- 18: $M = M \cup \{(j_i, j_i)\}$
- 19: **return** M

disambiguate these hypernyms using Algorithm 1. The sense vectors replace the aggregated context clues, so the stage described in Section 4.5 is superfluous for this option as well.

We also experimented in previous work with the construction of PCZs using this approach (Panchenko 2016), showing how to use sense embeddings for building PCZs, reaching satisfactory levels of recall and precision of matching as compared to a mapping defined by human judges.

While an empirical comparison of these options would be interesting, it is beyond the scope of this paper, where our main point is to demonstrate the benefits of linking manually created LRs with models induced by distributional semantics (by example of a count-based model).

5 Linking a proto-conceptualization to a lexical semantic resource

This section describes how a corpus-induced semantic network (a PCZ) is linked to a manually created semantic network, represented by an LR.

5.1 Linking induced senses to senses of the lexical resource

Now, we link each sense in our PCZ to the most suitable sense (if any) of an LR (see Figure 1 step 3). There exist many algorithms for knowledge base linking (Pavel and Euzenat 2013); here, we build upon simple, yet high-performing previous approaches to linking LRs that achieved state-of-the-art performance. These rely at their core on computing the overlap between the bags of words built from the LRs' concept lexicalizations, e.g. Navigli and Ponzetto (2012a) and Gurevych *et al.* (2012) (*inter alia*). Specifically, we develop (i) an iterative approach – so that the linking can benefit from the availability of linked senses from previous

iterations – (ii) leveraging the lexical content of the source and target resources. Algorithm 2 takes as input

- (1) a PCZ $T = \{(j_i, R_{j_i}, H_{j_i})\}$ where j_i is a sense identifier (i.e. mouse:1), R_{j_i} the set of its semantically related senses (i.e. $R_{j_i} = \{\text{keyboard:1}, \text{computer:0}, \dots\}$) and H_{j_i} the set of its hypernym senses (i.e. $H_{j_i} = \{\text{equipment:3}, \dots\}$);
- (2) an LR W : we experiment with: WordNet, a lexical database for English and BabelNet, a very large multilingual ‘encyclopedic dictionary’;
- (3) a threshold th over the similarity between pairs of concepts and a number m of iterations as a stopping criterion.

The algorithm outputs a mapping M , which consists of a set of pairs of the kind $(source, target)$ where $source \in T.\text{senses}$ is a sense of the input PCZ T and $target \in W.\text{senses} \cup source$ is the most suitable sense of W or $source$ when no such sense has been identified.

The algorithm starts by creating an empty mapping M (line 1). Then for each monosemous sense (e.g. Einstein:0 is the only sense in the PCZ for the term Einstein) it searches for a candidate monosemous sense (lines 2–6). If such monosemous candidate senses exist (line 4), we compare the two senses (line 5) with the following similarity function:

$$\text{sim}(j, c, M) = \frac{|T.\text{BoW}(j, M, W) \cap W.\text{BoW}(c)|}{|T.\text{BoW}(j, M, W)|}, \quad (3)$$

where

- (1) $T.\text{BoW}(j, M, W)$ is the set of words containing all the terms extracted from related/hypernym senses of j and all the terms extracted from the related/hypernym (i.e. already linked in M) synsets in W . For each synset from the LR, we use all synonyms and content words of the gloss.
- (2) $W.\text{BoW}(c)$ contains the synonyms and the gloss content words for the synset c and all the related synsets of c .

Then a new link pair (j_i, c_0) is added to M if the similarity score between j_i and c_0 meets or exceeds the threshold th (line 5). At this point, we collected a first set of disambiguated (monosemous) senses in M and start to iteratively disambiguate the remaining (polysemous) senses in T (lines 7–16). This iterative disambiguation process is similar to the one we described for the monosemous case (lines 2–6), with the main difference that, due to the polysemy of the candidates synsets, we instead use the similarity function to rank all candidate senses (lines 11–12) and select the top-ranked candidates for the mapping (lines 13–15). At the end of each iteration, we add all collected pairs to M (line 16). Finally, all unlinked j of T , i.e. induced senses that have no corresponding LR sense, are added to the mapping M (lines 17–18).

Comparison with other mapping algorithms. Previous work for the construction of BabelNet (Navigli and Ponzetto 2012a) and UBY (Gurevych *et al.* 2012) looked at the related problem of matching heterogeneous lexical semantic resources, i.e. Wikipedia and WordNet. In our scenario, however, we aim instead at establishing

Algorithm 3 Typing of the unmapped induced senses

Require: $M = (\text{source}, \text{target})$, W
Ensure: $H = (\text{source}, \text{type})$

```

1:  $H = \emptyset$ 
2: for all  $(\text{source}, \text{target}) \in M$  do
3:   if  $\text{target} \notin W$  then
4:      $\text{Rank} = 0$ 
5:     for all  $\text{related} \in R_{\text{source}}, \exists (\text{related}, \text{trelated}) \in M, \text{trelated} \in W$  do
6:       for all  $\text{hop} \in (1, 2, 3)$  do
7:         for all  $\text{ancestor} \in W.\text{ancestors}(\text{trelated}, \text{hop})$  do
8:            $\text{Rank}(\text{ancestor}) = \text{Rank}(\text{ancestor}) + 1.0/\text{hop}$ 
9:         for all  $\text{ntype} \in \text{Rank}.\text{top}(\text{top}_h)$  do
10:           $H = H \cup (\text{source}, \text{ntype})$ 
11: return  $H$ 

```

a bridge between any of these latter reference KBs and a PCZ – i.e. a fully disambiguated distributional semantic representation of distributionally induced word senses (cf. Section 4.5). Since we are working with a PCZ on the source side, as opposed to using Wikipedia, we cannot rely on graph-based algorithms such as PageRank (Niemann and Gurevych 2011) ‘out of the box’: while PCZs can be viewed as graphs, these are inherently noisy and require cleaning techniques in order to remove cycles and wrong relations (cf. Section 7 where we accordingly address the topic of taxonomy induction and cleaning within our framework). Similarly, the fact that PCZs are automatically induced from text – and hence potentially noisier than clean collaboratively generated content from Wikipedia – forces us to limit evidence for generating the mapping to local information, as opposed to, e.g. graph-based expansions used to boost the recall of BabelNet–WordNet mappings, cf. Navigli and Ponzetto (2012a). To overcome this ‘locality’ constraint, we develop an iterative approach to indirectly include non-local evidence based on previous mapping decisions. Our algorithm, in fact, uses previous mappings to additionally expand the bag-of-word of the candidate PCZ sense to be mapped, based on related/hypernym synsets linked in the previous iterations only (i.e. to keep the expansion ‘safe’, cf. Equation (3)).

5.2 Typing of the unmapped induced senses

An approach based on the bag-of-words from concept lexicalizations has the advantage of being simple, as well as high performing as we show later in the evaluation – cf. also findings from Navigli and Ponzetto (2012a). However, there could be still PCZ senses that cannot be mapped to the target LR, e.g. because of vocabulary mismatches, sparse concepts’ lexicalizations, or because they are simply absent in the resource.

Consequently, in the last phase of our resource creation pipeline, we link these ‘orphan’ PCZ senses (i.e. those from lines 17–18 of Algorithm 2), in order to obtain a unified resource, and propose a method to infer the type of those concepts that were not linked to the target LR. For example, so far we were not able to find a BabelNet sense for the PCZ item Roddenberry:10 (the author of ‘Star Trek’). However, by looking at the linked related concepts that share the same BabelNet hypernym – e.g. the PCZ items Asimov:3 *is-a* author_{BabelNet}, Tolkien:7 *is-a* author_{BabelNet}, Heinlein:8

Table 5. *Text corpora used in our experiments to induce distributional disambiguated semantic networks (proto-conceptualizations, PCZs)*

Name	Language	Source of texts	Genre	Size
Wiki News	English	Text of Wikipedia articles News articles: Gigaword and LCC	Encyclopedic Narrative, publicistic	35 million sent. 105 million sent.

is-a author_{BabelNet}, etc. – we can infer that Roddenberry:10 *is-a author:1*, since the latter was linked to the Babel synset *author_{BabelNet}*.

The input of Algorithm 3 consist of the mapping M of a PCZ to an LR W (cf. Algorithm 2). The output is a new mapping H containing pairs of the kind $(source, type)$ where $type$ is a type in W for the concept $source \in PCZ$. We first initialize the new mapping H as an empty set (line 1). Then for all the pairs $(source, target)$ where the target is a concept not included in the target LR W (line 3), we compute a rank of all the ancestors of each related sense that has a counterpart $trelated$ in W (lines 5–8). In other words, starting from linked related senses $trelated$, we traverse the taxonomy hierarchy (at most for three hops) in W and each time we encounter a sense $ancestor$ we increment its rank by the inverse of the distance to $trelated$. Finally, we add the pairs $(source, ntype)$ to H for all the $ntype$ at the top top_h in the $Rank$.

Finally, our final resource consists of (i) the PCZ; (ii) the mapping M of PCZ entries to the LR (e.g. WordNet or BabelNet); (iii) the mapping H of suggested types for the PCZ entries not mapped in M .

6 Experiments

In this section, we present results of four experiments, which intrinsically and extrinsically evaluate the quality of our HAR.

6.1 Corpora used for the induction of proto-conceptualizations

We evaluate our method using texts of different genres, namely standard newswire text vs. encyclopedic texts in order to examine performance in different settings. The corpora, described in Table 5, are a 105 million sentence news corpus composed of Gigaword (Parker *et al.* 2011) and the Leipzig Corpora Collection, (Richter *et al.* 2006)⁷ and a 35 million-sentence Wikipedia corpus⁸ from a 2011 dump.

We opt for these text collections because they were previously extensively used for the evaluation of distributional models based on the JoBimText framework (Biemann and Riedl 2013; Riedl and Biemann 2013). Specifically, previous work (Riedl and Biemann 2013) experimented with the induction of distributional models

⁷ <http://corpora.uni-leipzig.de>

⁸ The *wiki* corpus is downloadable (cf. Section 8). The *news* corpus is available by request due to license restrictions of the Gigaword corpus.

Table 6. Structural analysis of our five word sense inventories of the proto-conceptualizations (PCZs) used in our experiments

PCZ	Words			Senses		Polysemy		Rel.senses		Hyper.	
	n	#	Mono	Poly	#	Avg.	Max	#	Avg.	#	Avg.
News-p1.6	200	207 k	137 k	69 k	332 k	1.6	18	234 k	63.9	15 k	6.9
News-p2.3	50	200 k	99 k	101 k	461 k	2.3	17	298 k	44.3	15 k	5.8
Wiki-p1.8	200	206 k	120 k	86 k	368 k	1.8	15	300 k	59.3	15 k	4.4
Wiki-p6.0	30	258 k	44 k	213 k	1.5 M	6.0	36	811 k	16.9	52 k	1.7
Wiki-p1.6-mwe	200	465 k	288 k	176 k	765 k	1.6	13	662 k	46.6	30 k	3.2

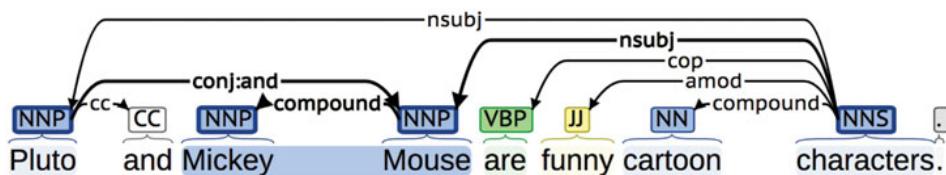


Fig. 4. (Colour online) Extraction of distributional dependency features for a multiword expression Mickey Mouse: all outgoing dependencies are used as features. This image was created using the Stanford dependency visualizer (<http://nlp.stanford.edu:8080/corenlp>).

on the basis of both corpora, and showed that the quality of semantic similarity (which, in turn, is used to build the DT, cf. Section 4.1) increases with corpus size. Since ‘more data’ helps, we experiment in this work with the full-sized corpora. Further description of the *wiki* and *news* text collections can be found in Riedl and Biemann (2013) and Riedl (2016, p. 94).

We experiment with different parameterizations of the sense induction algorithm to obtain PCZs with different average sense granularities, since *a priori*, there is no clear evidence for what the ‘right’ sense granularity of a sense inventory should be. Chinese Whispers sense clustering with the default parameters ($n = 200$) produced an average number of 2.3 (news) and 1.8 (wiki) senses per word with the usual power-law distribution of sense cluster sizes. Decreasing connectivity of the ego network via the n parameter leads to more fine-grained inventories (cf. Table 6).

Finally, we use the method described in Riedl and Biemann (2015) to compute a dataset that includes automatically extracted multiword terms using the Wikipedia corpus (*wiki-p1.6-mwe*). Since most of the multiwords are monosemous, average polysemy of this dataset decreased from 1.8 to 1.6 for the analogous model without multiwords (*wiki-p1.8*). To obtain a feature representation of a multiword expression, we gathered all outgoing dependency relations of this term as illustrated in Figure 4.

In Table 6, we present statistics for the five different resources we induce from our corpora. For each dataset, we report the counts of overall number of words (vocabulary size), including monosemous words and polysemous ones, respectively. For each PCZ, we report the cardinality, the average polysemy and the maximum polysemy. Finally, we report the overall and the average number of related senses

and hypernyms. Numbers vary across datasets due to the different nature of the two source corpora and the selection of different parameter values for sense induction.

While inducing senses directly from corpus data allows for large coverage and flexibility, it also makes it difficult to evaluate the quality of the resulting sense clusters (Agirre and Soroa 2007). Since we do not *a priori* know the sense granularity of the PCZs, the sense inventory input to our disambiguation and linking algorithms cannot be fixed in advance, e.g. in order to produce a static gold standard. Therefore, in our intrinsic evaluations (Sections 6.2–6.4), we assess the quality of our resources by manually validating a sample of the output of the different steps of our method. Later, in Section 6.5, we perform an extrinsic evaluation against a gold standard on a WSD benchmarking dataset from a SemEval task.

The PCZ described above were subsequently linked to WordNet 3.1 and BabelNet 2.5 using the method described above. All the models described above as well as the induced sense inventories and word similarity graphs can be accessed online (cf. Section 8).

6.2 Experiment 1: Quality of disambiguation of the related terms

Experimental setting. In this experiment, we evaluate the quality of Algorithm 1 for the disambiguation of related words (cf. Table 1) by performing a *post-hoc* evaluation using manual judgments on a sample of sense-disambiguated terms from one of our PCZ resources.

We manually selected a set of frequent nouns and proper nouns, such that each word has least two homonymous (as opposed to polysemous) word senses. We deliberately avoided words with polysemous senses, as word sense induction algorithms are known to robustly extract mostly coarse-grained inventories of homonymous words (Di Marco and Navigli 2013; Cecchini *et al.* 2017). The words were selected according to two criteria. First, each of the two homonymous word senses should have a comparable frequency – compare, for instance, the fairly common senses of *python* (animal) and *python* (language), as opposed to *boa* (animal) and *boa* (language), where the ‘language’ sense of the word *boa* is much rarer as compared to its ‘animal’ sense. Second, each of these senses should be common enough to be recognizable without the need of consulting a dictionary by a non-native, graduate-level speaker of English. We tested for sense frequencies and popularity by checking that selected senses were found among the top ones as listed in BabelNet. Using these criteria, we manually selected a set of seventeen nouns, such as *apple*, *java*, *python*, etc.⁹

Since our resources only partially overlap in terms of sense inventory, and there is no *a priori* reference sense granularity, we cannot perform evaluation on a shared

⁹ The full list of words, where senses are denoted in the brackets using the respective hypernyms: *apache* (tribe|software), *apple* (fruit|company), *bank* (river|institution), *commercial* (ad|business), *corvette* (car|ship), *jaguar* (animal|car), *java* (island|technology), *lotus* (flower|car), *mustang* (horse|car), *pascal* (person|language), *port* (sea-related|computer-related), *puma* (animal|brand), *python* (snake|language), *ruby* (gem|language), *sun* (star|company), *tiger* (animal|tank) and *viper* (snake|car).

Table 7. Accuracy of Algorithm 1 for disambiguation of related words evaluated on a set of seventeen frequent words each having two non-marginal homonymous word senses, e.g. as in *mouse* (*keyboard*) and *mouse* (*animal*)

Part of speech	# Word forms	# Senses	# Contexts	Accuracy
Nouns	15	30	1,055	0.94
Proper nouns	17	49	1,177	0.85
Adjectives	6	6	5,66	0.63
Verbs	4	6	86	0.76
All	42	91	2,284	0.84

gold standard. Consequently, we opt instead for a *post-hoc* evaluation of the accuracy of the disambiguation step, namely the fraction of correctly disambiguated related words among all disambiguated words. Post-hoc validations have major limitations in that they are time consuming, do not scale and hinder direct comparability across methods – nevertheless, they are commonly used in the field of knowledge acquisition to estimate the quality of knowledge resources (Banko *et al.* 2007; Suchanek *et al.* 2008; Carlson *et al.* 2010; Velardi, Faralli and Navigli 2013) (*inter alia*).

We performed manual validation as follows. We first collected all disambiguated entries of the *wiki-p1.6* model (cf. Table 1), where these seventeen target words appear and randomly sampled 15% of these entries to make annotation feasible, resulting in a dataset of 2,884 ambiguous-related words in context. We restrict evaluation to the *wiki-p1.6* model for two reasons: an encyclopedic source is expected to provide better sense coverage *a priori*, thus providing us with more evaluation instances, while a low number of clusters is in line with findings that graph-based sense induction methods can produce rather coarse high-quality clusterings (Cecchini *et al.* 2017).

Table 7 presents statistics of our dataset: note that we gathered word senses of all parts of speech that correspond to the selected seventeen words, including verbs and adjectives, for the sake of completeness of our study. An annotator with previous experience in lexicographic annotation performed the judgment of the 2,884 contexts in a curated way (using several rounds of feedback on random samples of annotations). The annotator was presented with a table containing four columns: (i) the target word, (ii) a sense cluster defining the sense of the target word, (iii) a sense cluster that defines the context of the target word. The last column collected the binary answer on the question whether the ‘definition’ cluster is semantically compatible with the ‘context’ cluster. Table 8 illustrates two examples of semantically compatible and incompatible clusters. The reasons of incompatibility of sense clusters are either the absence of obvious semantic relations between the words in the clusters (cf. the ‘planet’ vs. ‘basketball’ sense of *sun*) or simply incoherence of one or both sense clusters – i.e. the case when the annotator cannot figure out the meaning of the sense denoted by a cluster, such as the case for the context cluster of *tiger*. The annotator was instructed to consider a sense cluster to be interpretable if it was possible to grasp a dominant meaning by looking at the top twenty words, while allowing for a small fraction of spurious terms (since sense clusters are automatically generated).

Table 8. Examples provided to the assessor participating in the study with correct judgments

Word	Sense “Definition”	Sense “Context”	Related
Java	UNIX, Linux, Symbian, Unix, OS, Android, Mobile, Solaris, MS-DOS, Windows, iOS	screen, I/O, multiprocessor, IDE, repository, pak, Blu-ray, Graphics, Video, Itanium, ...	Yes
Python	hamster, lemurs, turtle, constrictor, lizard, orca, rhinoceros, cobra, ...	turtle, breeds, cattle, breed, cobra, Bulbul, Kingfisher, Mammals, starling, ...	Yes
Sun	Hearth, mirror, orb, soil, star, spotlight, temperature, water, solstice, burst, ...	eel, brave, Celtics, Wrangler, rockies, Chargers, Expos, Cavaliers, Cougars, padre, ...	No
Tiger	Macaque, deer, rhinoceros, Falcon, mascot, Whale, Gibbon, Hyena, boar, deer, ...	Nighthawk, Cessna, F-16, Valiant, Corsair, Maurer, Mirage, Reaper, Scorpion, ...	No

The subject was asked to determine if the first sense cluster (representing a sense definition of the ambiguous-related word) is semantically related to the second sense cluster (representing the context of the ambiguous-related word).

Results and discussion. The results of the experiment are summarized in Table 7.¹⁰ Performance of the disambiguation procedure for the proper names and nouns ranges from 0.85 to 0.94, thus indicating an overall high quality of the procedure. Note that the word senses of adjectives and verbs are mostly the result of part-of-speech tagging errors, since in the seed set of seventeen words, we added only nouns and proper nouns. Wrongly tagged words have in general more noisy, uninterpretable clusters.

To better understand the amount of spurious items in our sense clusters, we performed an additional manual evaluation where, for a sample of hundred randomly sampled noun PCZ items, we counted the ratio between wrong (e.g. rat for the computer sense of mouse) and correct (keyboard, computer, etc.) related sense that were found within the PCZs. We obtained a macro average of 0.0495 and a micro average of 0.0385 wrongly related senses within the PCZs. Moreover, eighty-three per cent of the above sample has no unrelated senses, and only two per cent have only a single unrelated sense with a macro average ratio between the wrong and correct related PCZs of 0.067. This indicates that, overall, the amount of spurious senses within clusters is indeed small, thus providing a high-quality context for an accurate disambiguation of noun DT clusters.

6.3 Experiment 2: Linking induced senses to lexical resources

Experimental setting. In this experiment, we evaluate the performance of our linking component (Section 5.1). For this, we choose two lexical-semantic networks:

¹⁰ The judgments are available for download (cf. Section 8).

WordNet (Fellbaum 1998), which has a high coverage on English common nouns, verbs and adjectives, and BabelNet (Navigli and Ponzetto 2012a), which also includes a large amount of proper nouns and senses gathered from multiple other sources, including Wikipedia.

We follow standard practices, e.g. Navigli and Ponzetto (2012a), and create five evaluation test sets, one for each dataset from Section 6.1, by randomly selecting a subset of 300 induced word senses for each dataset, and manually establishing a mapping from these senses to WordNet and BabelNet senses (senses that cannot be mapped are labeled as such in the gold standard).

We compare against the following two most frequent sense (MFS) baselines, which select from all the possible senses for a given term t :

- (1) The MFS in WordNet, where frequencies of senses are observed on a manually annotated semantic concordance (Miller *et al.* 1993).
- (2) The MFS in BabelNet. Since BabelNet combines WordNet and Wikipedia, this amounts to (i) the WordNet MFS for senses originally found in WordNet and (ii) the most cited (i.e. internally hyperlinked) Wikipedia page for senses derived from Wikipedia.

The quality and correctness of the mapping is estimated as accuracy on the ground-truth judgments, namely the amount of true mapping decisions among the total number of (potentially, empty) mappings in the gold standard. Each pair (j, c) in a mapping M created with Algorithm 2 is evaluated as (i) true positive (TP) when c is the most suitable sense in the LR for the induced word sense j ; (ii) true negative (TN) when c refers to j itself and there are no senses in the LR to capture the meaning expressed by j ; (iii) false positive (FP) when c is not the most suitable sense in the LR for the sense t ; (iv) false negative (FN) when c refers to j itself and there is a sense in the LR that captures the same meaning of j .

We also evaluate our mapping by quantifying coverage and extra-coverage on the reference resource:

$$\text{Coverage}(A, B) = \frac{|A \cap B|}{|B|} \quad \text{ExtraCoverage}(A, B) = \frac{|A/B|}{|B|} \quad (4)$$

where A is the set of LR synsets or induced word senses mapped in M using Algorithm 2, and B is the whole set of LR synsets. That is, Coverage indicates the percentage of senses of the LR sense inventory covered by the mapping M , whereas ExtraCoverage indicates the ratio of senses in M not linked to the LR sense inventory over the total number of senses in an LR. That is, ExtraCoverage is a measure of novelty to quantify the amount of senses discovered in T and not represented by the LR: it indicates the amount of ‘added’ knowledge we gain with our resource based on the amount of senses that cannot be mapped and are thus included as novel senses.

Table 9. Results on linking to lexical semantic resource: number of linked induced word senses, Coverage, ExtraCoverage, accuracy of our method and of the MFS baseline for our five datasets

PCZ	WordNet-linked				
	#linked senses	Cov.	ExtraCov.	Accuracy	MFS baseline
News-p1.6	88 k	34.5%	206.0%	86.9%	85.5%
News-p2.3	145 k	38.2%	267.0%	93.3%	85.0%
Wiki-p1.8	91 k	35.9%	234.7%	94.8%	80.5%
Wiki-p6.0	400 k	49.9%	919.9%	93.5%	74.2%
Wiki-mw-p1.6	81 k	30.7%	581.2%	95.3%	89.7%
BabelNet-linked					
	# linked senses	Cov.	ExtraCov.	Accuracy	MFS baseline
News-p1.6	164 k	1.3%	2.9%	81.8%	52.3%
News-p2.3	236 k	1.4%	3.9%	85.1%	57.2%
Wiki-p1.8	232 k	1.9%	2.4%	86.4%	41.0%
Wiki-p6.0	737 k	2.8%	1.3%	82.2%	54.7%
Wiki-mw-p1.6	589 k	4.7%	1.8%	83.8%	59.4%

Results and discussion. In Table 9, we present the results using the optimal parameter values (i.e. $th = 0.0$ and $m = 5$ of Algorithm 2).¹¹ For all datasets, the number of linked senses, Coverage and ExtraCoverage are directly proportional to the number of entries in the dataset – i.e. the finer the sense granularity, as given by a lower sense clustering n parameter, the lower the number of mapped senses, Coverage and ExtraCoverage.

In general, we report rather low coverage figures: the coverage in WordNet is always lower than fifty per cent (thirty per cent in one setting) and coverage in BabelNet is in all settings lower than five per cent. Low coverage is due to different levels of granularities between the source and target resource. Our target LRs, in fact, have very fine-grained sense inventories. For instance, BabelNet lists seventeen senses of the word *python* including two (arguably obscure ones) referring to particular roller coasters. In contrast, word senses induced from text corpora tend to be coarse and corpus specific. Consequently, the low coverage comes from the fact that we connect a coarse and a fine-grained sense inventory – cf. also previous work (Faralli and Navigli 2013) showing comparable proportions between coverage and extra-coverage of automatically acquired knowledge (i.e. glosses) from corpora.

Finally, our results indicate differences between the order of magnitude of the Coverage and ExtraCoverage when linking to WordNet and BabelNet. This high difference is rooted in the cardinality of the two sense inventories, whereas BabelNet encompasses millions of senses, WordNet contains hundreds of thousands – many

¹¹ To optimize m , we prototyped our approach on a dev set consisting of a random sample of 300 senses, and studied the curves for the number of linked induced senses to WordNet resp. BabelNet. The th value was then selected as to maximize the accuracy.

of them not covered in our corpora. Please note that an ExtraCoverage of about three per cent in BabelNet corresponds to about 300k novel senses. Overall, we take our results to be promising in that, despite the relative simplicity of our approach (i.e. almost parameter-free unsupervised linking), we are able to reach high accuracy figures in the range of around 87–95 per cent for WordNet and accuracies consistently above eighty per cent for BabelNet. This compares well against a random linking baseline that is able to achieve 44.2 per cent and 40.6 per cent accuracy on average when mapping to WordNet and BabelNet, respectively. Also, we consistently outperform the strong performance exhibited by the MFS baselines, which, in line with previous findings on similar tasks (Suchanek *et al.* 2008; Ponzetto and Navigli 2009) provide a hard-to-beat competitor. Thanks to our method, in fact, we are able to achieve an accuracy improvement over the MFS baseline ranging from 1.4 per cent to 14.3 per cent on WordNet mappings, and from 24.4 per cent to 45.4 per cent on BabelNet. Despite not being comparable, our accuracy figures are in the same ballpark as those reported by Navigli and Ponzetto (2012a) (cf. Table 1), who use a similar method for linking Wikipedia to WordNet.

Error analysis. To gain insights into the performance of our approach, as well as its limitations, we performed a manual error analysis of the output on the WordNet mappings, identifying a variety of sources of errors that impact the quality of the output resource. These include

- **part-of-speech tagging errors**, which may produce wrong senses such as non-existent ‘verbs’ (e.g. *tortilla:VB*) (about ten per cent of the errors);
- **Hearst patterns errors** that may extract wrong hypernyms such as *issue* for the entry *emotionalism* (about twenty per cent of the errors);
- **linking errors** where the accuracy strongly depends on the granularity of senses and relationships of the target LR (about seventy per cent of the errors).

More specifically, false positives are often caused by the selection of a synset that is slightly different from the most suitable one (i.e. semantic shift), whereas false negatives typically occur due to the lack of connectivity in the semantic network.

Even if the high values of the estimated accuracy (see Table 9) of our mapping approach indicate that we are generally performing well over all the classes of test examples (i.e. true positive, true negative, false positive and false negative), the performance figures exhibit a different order of magnitude between the count of true positives and true negatives. True negatives are senses in the ExtraCoverage that we estimate to be correct new senses not contained in the reference LR. For a sample of such senses, we performed an additional manual analysis, and identified the following reasons that explain our generally high ExtraCoverage scores:

- **Named entities and domain-specific senses** (about forty per cent of the true negatives): true negative senses are due to correct new senses not contained in the target LR. This holds in particular for WordNet, where encyclopedic content occurs in a spotty fashion in the form of a few examples for some classes.

Table 10. Statistics and performance on typing unmapped PCZ items: number of induced senses counting for ExtraCoverage, number of typed and untyped induced senses, accuracy of our method and accuracy of the MFS baseline for our five datasets

WordNet-linked					
PCZ	#extra senses	w types	w/o types	Accuracy	MFS baseline
News-p1.6	244 k	184 k	59 k	83.3%	80.7%
News-p2.3	316 k	226 k	90 k	91.4%	89.5%
Wiki-p1.8	277 k	225 k	51 k	89.2%	89.0%
Wiki-p6.0	1 M	675 k	487 k	81.2%	78.2%
Wiki-p1.6-mwe	683 k	538 k	144 K	78.8%	77.3%
BabelNet-linked					
PCZ	#extra senses	w types	w/o types	Accuracy	MFS baseline
News-p1.6	168 k	73 k	95 k	91.2%	87.2%
News-p2.3	225 k	89 k	135 k	90.3%	89.8%
Wiki-p1.8	208 k	143 k	65 k	87.2%	85.0%
Wiki-p6.0	1,4 M	278 k	1.1 M	41.2%	40.3%
Wiki-p1.6-mwe	552 k	342 k	209 k	89.6%	88.0%

- **Sense granularity misalignment** (about sixty per cent of the true negatives): true negatives that derive from excessively fine clustering, and should have been combined with other senses to represent a more generic sense.

6.4 Experiment 3: Typing of the unmapped induced senses

Experimental setting. The high ExtraCoverage rates from Section 6.3 show that our resource contains a large number of senses that are not contained in existing LRs such as WordNet and BabelNet. Besides, high accuracy scores in the evaluation of the quality of the sense clusters from Section 6.2 seem to indicate that such extra items are, in fact, of high quality. Crucially, for our purposes, information found among the extra coverage has enormous potential, e.g. to go beyond ‘Wikipedia-only’ sense spaces. Consequently, we next evaluate our semantic typing component (Section 5.2) to assess the quality of our method to include also these good extra clusters that, however, have no perfect mapping in the reference LR (WordNet, BabelNet).

Similarly to the experiments for the resource mapping (Section 6.3), we manually create five test sets, one for each dataset from Section 6.1, by randomly selecting 300 unmapped PCZ items for each dataset, and manually identifying the most appropriate type of each induced sense among WordNet or BabelNet senses. Given these gold standards, performance is then computed as standard accuracy on each dataset.

Results and discussion. In Table 10, we report the statistics and the estimated accuracy for the task of typing the previously unmapped senses found among

the ExtraCoverage. For each dataset and LR, we report the number of senses in the ExtraCoverage, the number of senses for which we inferred the type, the number of senses for which we were not able to compute a type, and the estimated accuracy for the types inferred by our method on the basis of either the links generated using our approach from Section 5.1, or those created using the MFS linking baseline. The results show that accuracy decreases for those datasets with higher polysemy. In particular, we obtain a low accuracy of 41.2 per cent for the ‘wiki-p6.0’ where the disambiguated thesaurus contains only a low number of related senses, resulting in sparsity issues. For the other settings, the accuracy ranges from 78.8 per cent to 91.4 per cent (WordNet) and from 87.2 per cent to 91.2 per cent (BabelNet). The MFS baseline accuracies of typing the unmapped induced senses (see Section 6.3) are lower, scoring 0.2 per cent to 2.7 per cent less accuracy for WordNet and 0.5 per cent to 4.2 per cent less accuracy for BabelNet: these results corroborate the previous ones on linking, where the MFS was shown to be a tough baseline. Besides, the higher performance figures achieved by the MFS on typing when compared to linking indicate that the typing task has a lower degree of difficulty in the respect that popular (i.e. frequent) types provide generally good type recommendations.

6.5 Experiment 4: Evaluation of enriched lexical semantic resources

Experimental setting. In our next experiment, we follow previous work (Navigli and Ponzetto 2012a) and benchmark the quality of our resources by making use of the evaluation framework provided by the SemEval-2007 task 16 (Cuadros and Rigau 2007) on the ‘Evaluation of wide-coverage knowledge resources’. This SemEval task is meant to provide an evaluation benchmark to assess wide-coverage LRs on the basis of a traditional lexical understanding task, namely WSD (Navigli 2009). The evaluation framework consists of the following two main phases:

- (1) **Generation of sense representations.** From each LR, sense representations, also known as ‘topic signatures’, are generated, which are sets of terms that are taken to be highly correlated with a set of target senses. In practice, a sense representation consists of a weighted vector, where each element corresponds to a term that is deemed to be related to the sense, and the corresponding weight quantifies its strength of association.
- (2) **WSD evaluation.** Next, sense representations are used as weighted bags of words in order to perform monolingual WSD using a Lesk-like method (cf. Lesk 1986) applied to standard lexical sample datasets. Given a target word in context and the sense representations for each of the target word’s senses, the WSD algorithm selects the sense with the highest lexical overlap (i.e. the largest number of words in common) between the sense representation and the target word’s textual context.

This SemEval benchmark utilizes performance on the WSD task as an indicator of the quality of the employed LR. This approach makes it possible to extrinsically compare the quality of different knowledge resources, while making as few assumptions as possible over their specific properties – this is because knowledge resources

Table 11. Sample entries of the hybrid aligned resource (HAR) for the words *mouse* and *keyboard*

PCZ ID	WordNet ID	PCZ related terms	PCZ context clues
mouse:0	mouse:wn1	rat:0, rodent:0, monkey:0, ...	rat:conj_and, gray:amod, ...
mouse:1	mouse:wn4	keyboard:1, computer:0, printer:0 ...	click:-prep_of, click:-nn,
keyboard:0	keyboard:wn1	piano:1, synthesizer:2, organ:0 ...	play:-dobj, electric:amod, ..
keyboard:1	keyboard:wn1	keypad:0, mouse:1, screen:1 ...	computer, qwerty:amod ...

Trailing numbers indicate sense identifiers. To enrich WordNet sense representations, we rely on related terms and context clues.

are simply viewed as sense representations, namely weighted bags of words. Besides, to keep the comparison fair, it uses a common and straightforward disambiguation strategy (i.e. Lesk-like word overlap) and a knowledge representation formalism (i.e. sense representations) that is equally shared across all LRs when evaluating them on the same reference dataset. Specifically, the evaluation is performed on two lexical sample datasets, from the Senseval-3 (Mihalcea, Chklovski and Kilgarriff 2004) and SemEval-2007 Task 17 (Pradhan *et al.* 2007) evaluation campaigns. The first dataset has coarse-grained and fine-grained sense annotations, while the second contains only fine-grained annotations. In all experiments, we follow the original task formulation and quantify WSD performance using standard metrics of recall, precision and balanced F-measure.

Here, we use the SemEval task to benchmark the ‘added value’ in knowledge applicable for WSD that can be achieved by enriching a standard resource like WordNet with disambiguated distributional information from our PCZs on the basis of our linking. To this end, we experiment with different ways of enriching WordNet-based sense representations with contextual information from our HAR. For each WordNet sense of a disambiguation target, we first build a ‘core’ sense representation from the content and structure of WordNet, and then expand it with different kinds of information that can be collected from the PCZ sense that is linked to it (cf. Table 11):

- **WordNet.** The baseline model relies solely on the WordNet LR. It builds a sense representation for each sense of interest by collecting synonyms and definition terms from the corresponding WordNet synset, as well as all synsets directly connected to it (we remove stop words and weigh words with term frequency).
- **WordNet + Related (news).** We augment the WordNet-based representation with related terms from the PCZ. That is, if the WordNet sense is linked to a corresponding induced sense in our resource, we add all related terms found in the linked PCZ sense to the sense representation.
- **WordNet + Related (news) + Context (news/wiki).** Sense representations of this model are built by taking the previously generated ones, and additionally including terms obtained from the context clues of either the news (+ Context (news)) or Wikipedia (+ Context (wiki)) corpora we use (see Section 6.1).

In the last class of models, we used up to 5,000 most relevant context clues per word sense. This value was set experimentally: performance of the WSD system gradually increased with the number of context clues, reaching a plateau at the value of 5,000. During aggregation, we excluded stop words and numbers from context clues. Besides, we transformed syntactic context clues to terms, stripping the dependency type, so they can be added to other lexical representations. For instance, the context clue `rat:conj_and` of the entry `mouse:0` was reduced to the feature `rat`.

Table 12 shows a complete example from our dataset that demonstrate how, thanks to our HAR, we are able to expand WordNet-based sense representations with many relevant terms from the related terms of our sense-disambiguated PCZs.

We compare our approach to four state-of-the-art systems: KnowNet (Cuadros and Rigau 2008), BabelNet, WN+XWN (Cuadros and Rigau 2007) and NASARI. KnowNet builds sense representations based on snippets retrieved with a web search engine. We use the best configuration reported in the original paper (KnowNet-20), which extends each sense with twenty keywords. BabelNet in its core relies on a mapping of WordNet synsets and Wikipedia articles to obtain enriched sense representations; here, we consider both original variants used to generate sense representations, namely collecting all BabelNet synsets that can be reached from the initial synset at distance one (BabelNet-1) or two (BabelNet-2) and then outputting all their English lexicalizations. The WN + XWN system is the top-ranked unsupervised knowledge-based system of Senseval-3 and SemEval-2007 datasets from the original competition (Cuadros and Rigau 2007). It alleviates sparsity by combining WordNet with the eXtended WordNet (Mihalcea and Moldovan 2001). The latter resource relies on parsing of WordNet glosses. For all these resources, we use the scores reported in the respective original publications.

NASARI provides hybrid semantic vector representations for BabelNet synsets, which are a superset of WordNet. Consequently, we follow a procedure similar to the one we use to expand WordNet-only sense representations with information from our PCZs – namely, for each WordNet-based sense representation, we add all features from the lexical vector of NASARI that corresponds to it.¹²

Thus, we compare our method to three hybrid systems that induce sense representations on the basis of WordNet and texts (KnowNet, BabelNet, NASARI) and one purely knowledge-based system (WN + XWN). Note that we do not include the supervised TSSEM system in this comparison, as in contrast to all other considered methods including ours, it relies on a large sense-labeled corpus.

Results and discussion. Table 13 presents results of the evaluation, which generally indicate the high quality of the sense representations in our HAR. Expanding WordNet-based sense representations with distributional information provides, in fact, a clear advantage over the original representation on both Senseval-3 and SemEval-2007 datasets. Using related words specific (via linking) to a given WordNet

¹² We used the version of lexical vectors (July 2016) featuring 4.4 million of BabelNet synsets, yet covering only seventy two per cent of word senses of the two datasets used in our experiments.

Table 12. *WordNet-only and PCZ-enriched sense representations for the fourth WordNet sense of the word *disk* (i.e. the ‘computer science’ one): the ‘core’ WordNet sense representation is additionally enriched with related words from our hybrid aligned resource*

Model	Sense representation
WordNet-only	memory, device, floppy, disk, hard, disk, disk, computer, science, computing, diskette, fixed, disk, floppy, magnetic, disc, magnetic, disk, hard, disc, storage, device
WordNet + Related (wiki)	recorder, disk, floppy, console, diskette, handset, desktop, iPhone, iPod, HDTV, kit, RAM, Discs, Blu-ray, computer, GB, microchip, site, cartridge, printer, tv, VCR, Disc, player, LCD, software, component, camcorder, cellphone, card, monitor, display, burner, web, stereo, internet, model, iTunes, turntable, chip, cable, camera, iphone, notebook, device, server, surface, wafer, page, drive, laptop, screen, pc, television, hardware, YouTube, dvr, DVD, product, folder, VCR, radio, phone, circuitry, partition, megabyte, peripheral, format, machine, tuner, website, merchandise, equipment, gb, discs, MP3, hard-drive, piece, video, storage device, memory device, microphone, hd, EP, content, soundtrack, webcam, system, blade, graphic, microprocessor, collection, document, programming, battery, keyboard, HD, handheld, CDs, reel, web, material, hard-disk, ep, chart, debut, configuration, recording, album, broadcast, download, fixed disk, planet, pda, microfilm, iPod, videotape, text, cylinder, cpu, canvas, label, sampler, workstation, electrode, magnetic disc, catheter, magnetic disk, Video, mobile, cd, song, modem, mouse, tube, set, ipad, signal, substrate, vinyl, music, clip, pad, audio, compilation, memory, message, reissue, ram, CD, subsystem, hdd, touchscreen, electronics, demo, shell, sensor, file, shelf, processor, cassette, extra, mainframe, motherboard, floppy disk, lp, tape, version, kilobyte, pacemaker, browser, Playstation, pager, module, cache, DVD, movie, Windows, cd-rom, e-book, valve, directory, harddrive, smartphone, audiotape, technology, hard disk, show, computing, computer science, Blu-Ray, blu-ray, HDD, HD-DVD, scanner, hard disc, gadget, booklet, copier, playback, TiVo, controller, filter, DVDs, gigabyte, paper, mp3, CPU, dvd-r, pipe, cd-r, playlist, slot, VHS, film, videocassette, interface, adapter, database, manual, book, channel, changer, storage

sense provides substantial improvements in the results. Further expansion of sense representations with context clues (cf. Table 1) provides a modest improvement on the SemEval-2007 dataset only. Consequently, the results seem to indicate that context clues generally do not provide additional benefits over the expansions provided from the related terms of the linked PCZ items for task of generating sense representations.

On the Senseval-3 dataset, our hybrid models show better performance than all unsupervised knowledge-based approaches considered in our experiment. On the

Table 13. Comparison of our approach to the state of the art unsupervised knowledge-based methods on the SemEval-2007 Task 16 (weighted setting)

Model	Senseval-3 fine-grained			SemEval-2007 fine-grained		
	Precision	Recall	F ₁	Precision	Recall	F ₁
Random	19.1	19.1	19.1	27.4	27.4	27.4
WordNet (WN)	29.7	29.7	29.7	44.3	21.0	28.5
WN + Related (news)	47.5	47.5	47.5	54.0	50.0	51.9
WN + Related (news) + Context (news)	47.2	47.2	47.2	54.8	51.2	52.9
WN + Related (news) + Context (wiki)	46.9	46.9	46.9	55.2	51.6	53.4
BabelNet-1	44.3	44.3	44.3	52.2	46.3	49.1
BabelNet-2	35.0	35.0	35.0	56.9	53.1	54.9
KnowNet	44.1	44.1	44.1	49.5	46.1	47.7
NASARI (lexical vectors)	32.3	32.2	32.2	49.3	45.8	47.5
WN + XWN	38.5	38.0	38.3	54.9	51.1	52.9

The best results per section (i.e. the ones using our resources vs. those from the previous literature) are boldfaced, the best results overall are underlined.

SemEval-2007 dataset instead, we perform on a par, yet slightly below BabelNet’s best setting. Error analysis of the sense representations suggests that the extra performance of BabelNet on the SemEval data seems to derive from an aggressive graph-based expansion technique that leverages semantic relations harvested from Wikipedias in many languages – cf. the overall lower performance obtained by collecting sense representations from all Babel synsets at depth 1 only (BabelNet-1) vs. those that can be reached with two hops (BabelNet-2). This ‘joint multilingual’ approach has also been shown to benefit WSD in general (Navigli and Ponzetto 2012b), and represents an additional source of semantic information not present in our resource (we leave multilinguality for future work).

The results generally indicate the high quality of our HAR in a downstream application scenario, where we show competitive results while being much less resource intensive. This is because our method relies only on a relatively small LR like WordNet and raw, unlabeled text, as opposed to huge LRs like BabelNet or KnowNet. That is, while our method shows competitive results better or on a par with other state-of-the-art systems, it does not require access to web search engines (KnowNet), the structure and content of a very large collaboratively generated resource and texts mapped to its sense inventory (BabelNet, NASARI), or even a machine translation system or multilingual interlinked Wikipedias (BabelNet).

Related work on unsupervised WSD using the induced sense inventory. This article is focused on the HAR, e.g. a PCZ linked to an LR, as in the *knowledge-based* WSD experiment described above. However, the induced sense inventory is a valuable resource on its own and can be used to perform *unsupervised knowledge-free* WSD. In this case, the induced sense representations, featuring context clues, related words

and hypernyms, are used as features representing the induced senses. In our related experiments with sparse count-based (Panchenko *et al.* 2016, 2017c) and dense prediction-based (Pelevina *et al.* 2016) distributional models, we show that such unsupervised knowledge-free disambiguation models yield state-of-the-art results as compared to the unsupervised systems participated in SemEval 2013 and the AdaGram (Bartunov *et al.* 2016) sense embeddings model. An interactive demo that demonstrates our model developed in these experiments is described in Figure 2 and in Panchenko *et al.* (2017b).

7 Applications

Linked distributional disambiguated resources carry a great potential to positively impact many knowledge-rich scenarios. In this section, we leverage our resource for a few downstream applications of knowledge acquisition, namely (i) noise removal in automatically acquired knowledge graphs and (ii) domain taxonomy induction from scratch.

7.1 Linking knowledge resources helps taxonomy construction

Task definition. We examine a crucial task in learning a taxonomy (i.e. the *isa* backbone of a lexical-semantic resource) from scratch (Bordea *et al.* 2015, 2016), namely the induction of clean taxonomic structures from noisy hypernym graphs such as, for instance, those obtained from the extractions of hyponym–hypernym relations from text. In this task, we are given as input a list of subsumption relations between terms or, optionally, word senses – e.g. those from our PCZs (Figure 1) – which can be obtained, for instance, by exploiting lexical-syntactic paths (Hearst 1992; Snow, Jurafsky and Ng 2004), distributional representations of words (Baroni *et al.* 2012; Roller, Erk and Boleda 2014) or a combination of both (Shwartz *et al.* 2016). Due to the automatic acquisition process, such lists typically contain noisy, inconsistent relations – e.g. multiple inheritances and cycles – which do not conform to the desired, clean hierarchical structure of a taxonomy. Therefore, the task of *taxonomy construction* focuses on bringing order among these extractions, and on removing noise by organizing them into a directed acyclic graph (Kozareva and Hovy 2010).

Related work. State-of-the-art algorithms differ by the amount of human supervision required and their ability to respect some topological properties while pruning the noise. Approaches like those of Kozareva and Hovy (2010), Velardi *et al.* (2013) and Kapanipathi *et al.* (2014), for instance, apply different topological pruning strategies that require to specify the root and leaf concept nodes of the KB in advance – i.e. a predefined set of abstract top-level concepts and lower terminological nodes, respectively. The approach of Faralli, Stilo and Velardi (2015) avoids the need of such supervision with an iterative method that uses an efficient variant of topological sorting (Tarjan 1972) for cycle pruning. Such lack of supervision, however, comes at the cost of not being able to preserve the original connectivity between the

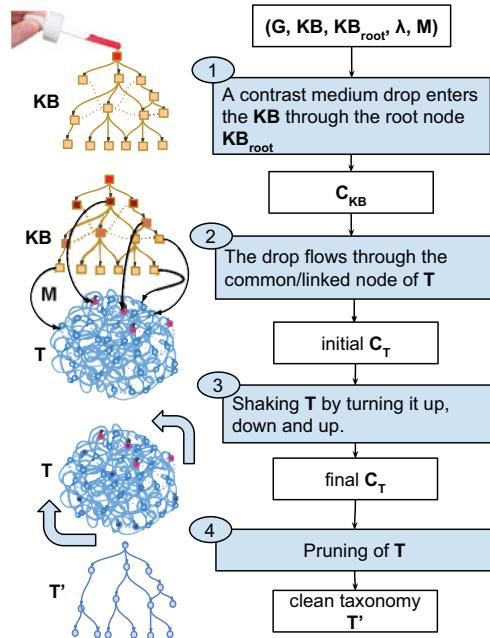


Fig. 5. (Colour online) The ContrastMedium (CM) algorithm for taxonomy construction.

top (abstract) and the bottom (instance) concepts. Random edge removal (Faralli *et al.* 2015), in fact, can lead to disconnected components, a problem shared with the OntoLearn Reloaded approach (Velardi *et al.* 2013), which cannot ensure such property when used to approximate a solution on a large noisy graph.

ContrastMedium algorithm. Links between heterogeneous knowledge resources, like those found within our HAR (Section 5), can be leveraged, together with a specialized algorithm, in order to advance the state of the art in taxonomy construction. To this end, we use ContrastMedium, a novel algorithm (Faralli *et al.* 2017) that is able to extract a clean taxonomy from a noisy knowledge graph without needing to know in advance – that is, having to manually specify – the top-level and leaf concepts of the taxonomy, while preserving the overall connectivity of the graph. ContrastMedium achieves this by projecting the taxonomic structure from a *reference taxonomy* (e.g. WordNet or the taxonomic *isa* backbone of BabelNet) onto a *target (noisy) hypernym graph* – for instance, the graph built from the set of hypernym relations in our HAR (Section 3) – on the basis of links found between the two resources, e.g. those automatically generated using our method from Section 5.

Metaphorically, in the context of clinical analysis, a contrast medium (CM) is injected into the human body to highlight specific complex internal body structures (in general, the cardiovascular system). In a similar fashion, our approach, which is summarized in Figure 5, starts by detecting the topological structure of the reference taxonomy by propagating a certain amount of CM that we initially inject through its root node (step 1). The highlighted structure indicates the distance of a node with respect to the root, with the lowest values of CM indicating the leaf terminological

nodes. The observed quantities are then transferred to corresponding nodes of the target hypernym graph by following the links between the two resources (step 2). Next, the medium is propagated by ‘shaking’ the noisy graph. We let the fluid reach all its components by alternating two phases of propagation: letting the CM flow via both incoming (shake up) and outgoing (shake down) edges (step 3). Finally, we use the partial order induced by the level of CM observed in each node to drive the pruning phase, and we ‘stretch’ the linked noisy knowledge graph into a proper taxonomy, namely a directed acyclic graph (step 4).

Evaluation. We benchmark ContrastMedium by comparing the quality of its output taxonomies against those obtained with the state-of-the-art approach of Faralli *et al.* (2015). The latter relies on Tarjan’s topological sorting, which iteratively searches for a cycle (until no cycle can be found) and randomly removes an edge from it. We applied the two approaches to our linked resources and evaluated the performance on a three-way classification task to automatically detect the level of granularity of a concept. Pruning accuracy is estimated on the basis of a dataset of ground-truth judgments that were created using double annotation with adjudication from a random sample of 1,000 nodes for each noisy hypernym graph ($\kappa = 0.657$ (Fleiss 1971)). To produce a gold-standard, coders were asked to classify concepts from the random sample as (i) a root, top-level abstract concept – i.e. any of entity, object, etc. and more in general nodes that correspond to abstract concepts that we can expect to be part of a core ontology such as, for instance, DOLCE (Gangemi *et al.* 2002); (ii) a leaf terminological node (i.e. instances such as Pet Shop Boys) or (iii) a middle-level concept (e.g. celebrity), namely concepts not fitting into any of the previous classes.

We compute standard accuracy for each of the three classes. That is, we compare the system outputs against the gold standards and obtain three accuracy measures: one for the root nodes (A_R), one for the nodes ‘in the middle’ (A_M) and finally one for the leaf nodes (A_L). In Table 14, we show some of the results of the evaluation. Thanks to ContrastMedium, we are able to achieve, even despite the baseline already reaching very high performance levels (well above ninety per cent accuracy), improvements of up to six percentage points, with an overall error reduction between around forty per cent and sixty per cent. This performance improvements are due to the fact that ContrastMedium is able to (i) identify important topological clues among ground-truth taxonomic relations from the reference taxonomy and (ii) project them onto the noisy graph on the basis of the links found in the mapping between the two resources. That is, the availability of a mapping between knowledge resources helps us to project the supervision information from the clean source taxonomy into the target noisy graph without the need of further supervision. The reference taxonomy provides us with ground-truth taxonomic relations – this renders our method as knowledge-based, not knowledge-free. However, the availability of resources like, for instance, WordNet for English or the multilingual BabelNet implies that these requirements are nowadays trivially satisfied. The mapping, in turn can be automatically generated with high precision using any of the existing solutions for KB mapping, e.g. our algorithm from Section 5, or by relying on

Table 14. Pruning accuracy of the CM

	ContrastMedium		
	A_R	A_M	A_L
News-p1.6	98.9%	98.3%	99.3%
News-p2.3	98.7%	98.7%	99.9%
Wiki-p1.8	97.6%	94.7%	97.3%
Wiki-p6.0	95.9%	94.3%	98.3%
	Tarjan (baseline)		
	A_R	A_M	A_L
News-p1.6	93.3%	94.6%	95.3%
News-p2.3	95.7%	94.7%	95.6%
Wiki-p1.8	93.1%	87.3%	94.1%
Wiki-p6.0	89.5%	90.1%	92.8%

ground-truth information from the Linguistic Linked Open Data cloud (Chiarcos, Hellmann and Nordhoff 2012).

What is perhaps the most interesting bit in our approach is the fact that by combining our unsupervised framework for knowledge acquisition from text (Section 3) with ContrastMedium, we are able to provide an *end-to-end solution for high-quality, unsupervised taxonomy induction from scratch*, i.e. without any human effort.

7.2 Inducing taxonomies from scratch using the hybrid aligned resource

Task definition. We now look at how ContrastMedium can be used as component within a larger system to enable end-to-end taxonomy acquisition from text. In general, taxonomy learning from scratch (Bordea *et al.* 2015, 2016) consists of the task of inducing a hypernym hierarchy from text alone (Biemann 2005), this typically starts with an initial step of finding hypernymy relations from texts, which is followed by a taxonomy construction phase in which local semantic relations are arranged together within a proper global taxonomic structure (cf. previous Section 7.1).

Related work. Existing approaches like Kozareva and Hovy (2010) (among others) use Hearst-like patterns (Hearst 1992) to bootstrap the extraction of terminological sister terms and hypernyms. Instead, in Velardi *et al.* (2013), the extraction of hypernymy relations is performed with a classifier, which is trained on a set of manually annotated definitions from Wikipedia (Navigli and Velardi 2010), being able to detect definitional sentences and to extract the definiendum and the hypernym. In these systems, the harvested hypernymy relations are then arranged into a taxonomy structure, e.g. using cycle pruning and ‘longest path’ heuristics to

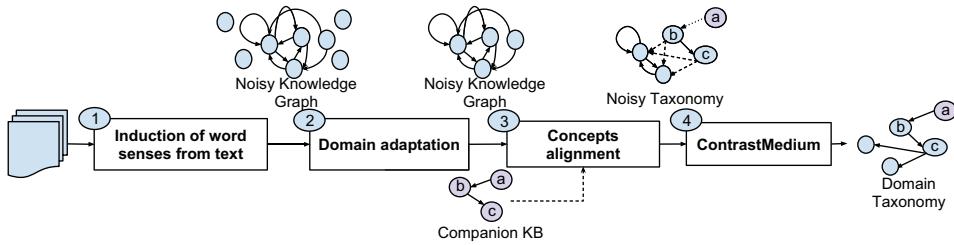


Fig. 6. (Colour online) Our full end-to-end pipeline for taxonomy induction from scratch.

induce a directed acyclic graph structure (Kozareva and Hovy 2010) or by relying on a variant of Chu-Liu Edmonds' optimal branching algorithm (Velardi *et al.* 2013). In general, all such lexical-based approaches suffer from the limitation of not being sense aware, which results in spurious taxonomic structures. Now, our fully disambiguated sense inventories could potentially overcome this problem and enable to a step forward toward the induction of high-quality, full-fledged taxonomies. In fact, we now show how the linked/semantic nature of our resources enables the development of a complete approach for taxonomy induction from scratch that achieves state-of-the-art performance with virtually no explicit supervision.

Using HAR to learn taxonomies from scratch. We now focus on the task of taxonomy induction by exploiting our HAR. Our approach is based on a five-stages pipeline as follows (see Figure 6):

- (1) Create PCZs as described in Section 4.
- (2) Filter out-of-domain concepts from our PCZs on the basis of domain-terminology-based heuristics. We construct domain-specific PCZs for a target domain by a simple lexical filtering. First, we build an extended lexicon of each domain on the basis of a seed vocabulary of the domain – i.e. domain terminologies such as those provided from the TExEval challenge (see below). Namely, for each seed term, we retrieve all semantically similar terms on the basis of the PCZ.
- (3) Build an HAR by linking the PCZs to a companion taxonomy (e.g. WordNet, BabelNet, etc.) based on the methods from Section 5.
- (4) Build a noisy hypernym graph by taking the union of the hypernym relations found within our PCZs.
- (5) Apply ContrastMedium (Section 7.1) to remove noise from the graph and obtain a proper taxonomic structure.

That is, the combination of all our methods we presented so far provides us with a full end-to-end pipeline for taxonomy induction from scratch. Arguably, our approach is unsupervised in that it does not require any explicit human effort other than the knowledge encoded within the reference LRs. Both PCZs and links to reference knowledge resources are automatically induced in an unsupervised way. Moreover, links to existing LRs, as used in ContrastMedium, provide us with a source of knowledge-based supervision that is leveraged to clean PCZs and turn them into full-fledged taxonomies. More precisely, our framework is fully unsupervised up to

the linking part. However, unsupervised linking to a knowledge base and using the knowledge base for taxonomy construction indeed requires the knowledge base itself. To this end, we use freely available resources like WordNet and BabelNet. Given the linking, we can exploit the knowledge from these LRs, together with a knowledge-based method (ContrastMedium), without the need for additional human effort or supervision. That is, the fact that these lexical knowledge resources already exist and are publicly available implies that we can apply our framework with no extra human intervention.

Experiments. We use the evaluation benchmark from the most recent edition of the TExEval challenge (SemEval 2016 – task 13) (Bordea *et al.* 2016). Our experimental setting consists of the following components:

- **Three gold-standard taxonomies**, namely the FOOD’s sub hierarchy of the Google products taxonomy,¹³ as well as the NASEM¹⁴ and EuroVoc¹⁵ taxonomies of SCIENCE.
- **The task baseline**, which induces the taxonomy structure only from relations between compound terms such as juice, apple juice by applying simple substring inclusion heuristics. This baseline approach does not leverage any external or background knowledge and only uses the input domain terminology.
- **The Cumulative Fowlkes&Mallows evaluation measure** (CF&M): this enables the comparison of a system taxonomy against a gold standard at different levels of depth of the taxonomy, as obtained by penalizing errors at the highest cuts of the hierarchy (Velardi *et al.* 2012).
- **The task participant’s systems**: (1) the JUNLP system (Maitra and Das 2016) makes use of two string inclusion heuristics combined with information from BabelNet; (2) the NUIG–UNLP system (Pocostales 2016) implements a semi-supervised method that finds hypernym candidates by representing them as distributional vectors (Mikolov, Yih and Zweig 2013); (3) the QASSIT system (Cleuziou and Moreno 2016) is a semi-supervised methodology for the acquisition of lexical taxonomies based on genetic algorithms. It is based on the theory of pretopology (Gil-Lafuente and Aluja 2012) that offers a powerful formalism to model semantic relations; (4) our task-winning TAXI system (Panchenko *et al.* 2016) that relies on combining two sources of evidence: substring matching and Hearst-like patterns. Hypernymy relations are extracted from Wikipedia, GigaWord, ukWaC, a news corpus and the CommonCrawl, as well as from a set of focused crawls; (5) the USAAR system (Tan, Bond and van Genabith 2016) exploits the hypernym endo/exocentricity (Brugmann 1904) as a practical property for hypernym identification.

¹³ <http://www.google.com/basepages/producttype/taxonomy.en-US.txt>

¹⁴ http://sites.nationalacademies.org/PGA/Resdoc/PGA_0445

¹⁵ <http://eurovoc.europa.eu/drupal/>

Table 15. Comparison based on the SemEval 2016 task 13 benchmark for FOODS and SCIENCES domains. We report the Cumulative Fowlkes&Mallows measure

System	Google	NASEM	EuroVoc
	FOODS	SCIENCES	SCIENCES
Baseline	0.0019	0.0163	0.0056
JUNLP	0.2608	0.1774	0.1373
NUIG-UNLP	–	0.0090	0.1517
QASSIT	–	0.5757	0.3893
TAXI	0.2021	0.3634	0.3893
USAAR	0.0000	0.0020	0.0023
WordNet	0.5870	0.5760	0.6243
Our approach	0.6862	0.7000	0.8157

- **A reference taxonomy:** we use WordNet for evaluation purposes by treating it the same way as any other participant system's output.

In Table 15, we report the results on the SemEval gold standards. Our approach significantly (χ^2 test, $p < 0.01$) outperforms all the other systems in all domains (i.e. Google FOOD, NASEM SCIENCE and EuroVoc SCIENCE), as well as the ground-truth taxonomy provided by WordNet. More importantly, the results indicate the overall robustness of our approach; that is, leveraging distributional semantics and symbolic knowledge (i.e. through linking to reference LRs) together is able to outperform not only the WordNet gold standard, which has limited coverage for fine-grained specific domains like these, but also the SemEval task participants, which all rely in some way or another on simple, yet powerful substring heuristics.

8 Conclusions

We have presented a framework for enriching lexical semantic resources, such as WordNet, with distributional information. Lexical semantic resources provide a well-defined semantic representation, but typically contain no corpus-based statistical information and are static in nature. Distributional semantic methods are well suited to address both of these problems, since models are induced from (in-domain) text and can be used to automatize the process of populating ontologies with new concepts. However, distributional semantic representations based on dense vectors have also major limitations in that they are uninterpretable on the symbolic level. By linking these representations to a reference LR, we can interpret them in an explicit way by means of the underlying relational knowledge model.

We provided a substantial investigation on enrichment of LRs with distributional semantics and evaluated the results in intrinsic and extrinsic ways showing that the resulting hybrid sense representations can be successfully applied to a variety of tasks that involve lexical-semantic knowledge. We tested the quality of the hybrid resources generated by our framework with a battery of intrinsic evaluations. Additionally,

we benchmarked the quality of our resource in a knowledge-based WSD setting, where we showed that our arguably low-resource approach – in that we rely only on a small LR like WordNet and raw, unlabeled text – reaches comparable quality with BabelNet, which in contrast is built on top of large amounts of high-quality collaborative content from Wikipedia. Finally, by combining distributional semantic vectors with links to a reference LRs, we are able to pave the way to the development of new algorithms to tackle hard, high-end tasks in knowledge acquisition like taxonomy cleaning and unsupervised, end-to-end taxonomy learning.

We believe that the hybrid LRs developed in our work will benefit high-end applications, e.g. ranging from entity-centric search (Lin *et al.* 2012; Schuhmacher, Dietz and Ponzetto 2015) all the way through full-fledged document understanding (Rospocher *et al.* 2016).

Downloads. We release all resources produced in this work under CC-BY 4.0 License¹⁶: (i) the PCZs resulting from our first experiment (Section 6.2); (ii) following the guidelines in McCrae, Fellbaum and Cimiano (2014), we created an RDF representation to share the mapping between our PCZs and lexical knowledge graphs (i.e. WordNet and BabelNet) (see Section 6.3) in the Linked Open Data Cloud; (iii) the types of the unmapped PCZ senses produced in the third experiment (see Section 6.4). All datasets, evaluation judgments, source code, and the demo can be accessed via <http://web.informatik.uni-mannheim.de/joint>.

References

- Agirre, E., and Soroa, A. 2007. SemEval-2007 Task 02: evaluating word sense induction and discrimination systems. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval'07)*, Prague, Czech Republic.
- Aprosio, A. P., Giuliano, C., and Lavelli, A. 2013. Extending the coverage of DBpedia properties using distant supervision over Wikipedia. In *Proceedings of the 2013 International Conference on NLP and DBpedia – Volume 1064 (NLP-DBPEDIA'13)*, Sydney, Australia.
- Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., and Etzioni, O. 2007. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, Hyderabad, India.
- Baroni, M., Bernardi, R., Do, N.-Q., and Shan, C.-C. 2012. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL'12)*, Avignon, France.
- Baroni, M., Dinu, G., and Kruszewski, G. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL'14)*, Baltimore, MD, USA.
- Bartunov, S., Kondrashkin, D., Osokin, A., and Vetrov, D. P. 2016. Breaking sticks and ambiguities with adaptive skip-gram. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS'16)*, Cadiz, Spain.
- Biemann, C. 2005. Ontology learning from text: a survey of methods. *LDV Forum* **20**(2): 75–93.

¹⁶ <https://creativecommons.org/licenses/by/4.0/>

- Biemann, C. 2006. Chinese whispers: an efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing, TextGraphs-1*, Brooklyn, NY, USA.
- Biemann, C., and Riedl, M. 2013. Text: now in 2D! A framework for lexical expansion with contextual similarity. *Journal of Language Modelling* 1(1): 55–95.
- Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., and Hellmann, S. 2009. DBpedia – a crystallization point for the web of data. *Journal of Web Semantics* 7(3): 154–65.
- Bordea, G., Buitelaar, P., Faralli, S., and Navigli, R. 2015. SemEval-2015 task 17: taxonomy extraction evaluation (TExEval). In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval@NAACL-HLT'15)*, Denver, CO, USA.
- Bordea, G., Lefever, E., and Buitelaar, P. 2016. SemEval-2016 task 13: taxonomy extraction evaluation (TExEval-2). In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval@NAACL-HLT'16)*, San Diego, CA, USA.
- Bordes, A., Weston, J., Collobert, R., and Bengio, Y. 2011. Learning structured embeddings of knowledge bases. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI'11)*, San Francisco, CA, USA.
- Brugmann, K. 1904. *Kurze vergleichende Grammatik der indogermanischen Sprachen*. Strassburg, France: Karl J. Trübner.
- Bryl, V., and Bizer, C. 2014. Learning conflict resolution strategies for cross-language Wikipedia data fusion. In *Proceedings of the 23rd International Conference on World Wide Web (WWW'14 Companion)*, Seoul, South Korea.
- Camacho-Collados, J., Pilehvar, M. T., and Navigli, R. 2015a. A unified multilingual semantic representation of concepts. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Volume 1: Long Papers (ACL'15)*, Beijing, China.
- Camacho-Collados, J., Pilehvar, M. T., and Navigli, R. 2015b. NASARI: a novel approach to a semantically aware representation of items. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'15)*, Denver, CO, USA.
- Carlson, A., Betteridge, J., Kisiel, R., Settles, B., Hruschka, Jr., E. R., and Mitchell, T. M. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI'10)*, Atlanta, GA, USA.
- Cecchini, F. M., Riedl, M., and Biemann, C. 2017. Using pseudowords for algorithm comparison: an evaluation framework for graph-based word sense induction. In *Proceedings of the 21st Nordic Conference on Computational Linguistics (NODALIDA'17)*, Gothenburg, Sweden.
- Chen, X., Liu, Z., and Sun, M. 2014. A unified model for word sense representation and disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, A meeting of SIGDAT, a Special Interest Group of the ACL (EMNLP'14)*, Doha, Qatar.
- Chiarcos, C., Hellmann, S., and Nordhoff, S. 2012. Linking linguistic resources: examples from the open linguistics working group. In C. Chiarcos, S. Nordhoff, and S. Hellmann (eds.), *Linked Data in Linguistics – Representing and Connecting Language Data and Language Metadata*, pp. 201–16. Heidelberg, Germany: Springer.
- Clark, S. 2015. Vector space models of lexical meaning. In S. Lappin, and C. Fox (eds.), *Handbook of Contemporary Semantics*, 2nd edition, pp. 493–522. New York, NY, USA: Wiley-Blackwell.
- Cleuziou, G., and Moreno, J. G. 2016. QASSIT at SemEval-2016 Task 13: on the integration of semantic vectors in pretopological spaces for lexical taxonomy acquisition. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval@NAACL-HLT'16)*, San Diego, CA, USA.

- Cuadros, M., and Rigau, G. 2007. SemEval-2007 task 16: evaluation of wide coverage knowledge resources. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval'07)*, Prague, Czech Republic.
- Cuadros, M., and Rigau, G. 2008. KnowNet: building a large net of knowledge from the web. In *Proceedings of the 22nd International Conference on Computational Linguistics – Volume 1 (COLING'08)*, Manchester, UK.
- Curran, J. R. 2002. Ensemble methods for automatic thesaurus extraction. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing – Volume 10 (EMNLP'02)*, Philadelphia, PA, USA.
- De Marneffe, M.-C., MacCartney, B., and Manning, C. D. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy.
- Di Marco, A., and Navigli, R. 2013. Clustering and diversifying web search results with graph-based word sense induction. *Computational Linguistics* 39(3): 709–54.
- Evert, S. 2005. *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. Ph.D. thesis. Institut für maschinelle Sprachverarbeitung, University of Stuttgart, Germany.
- Fader, A., Soderland, S., and Etzioni, O. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'11)*, Edinburgh, UK.
- Faralli, S., and Navigli, R. 2012. A new minimally supervised framework for domain word sense disambiguation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL'12)*, Jeju Island, South Korea.
- Faralli, S., and Navigli, R. 2013. Growing multi-domain glossaries from a few seeds using probabilistic topic models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, A meeting of SIGDAT, a Special Interest Group of the ACL (EMNLP'13)*, Seattle, WA, USA.
- Faralli, S., Panchenko, A., Biemann, C., and Paolo Ponzetto, S. 2017. The contrast medium algorithm: taxonomy induction from noisy knowledge graphs with just a few links. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, Volume 1: Long Papers (EACL'17)*, Valencia, Spain.
- Faralli, S., Stilo, G., and Velardi, P. 2015. Large scale homophily analysis in twitter using a twixonomy. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*, Buenos Aires, Argentina.
- Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E. H., and Smith, N. A. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'15)*, Denver, CO, USA.
- Faruqui, M., and Kumar, S. 2015. Multilingual open relation extraction using cross-lingual projection. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'15)*, Denver, CO, USA.
- Fellbaum, C. (ed.) 1998. *WordNet: An Electronic Database*. Cambridge, MA: MIT Press.
- Fleiss, J. L. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin* 76(5): 378.
- Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., and Schneider, L. 2002. Sweetening ontologies with DOLCE. In *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web: 13th International Conference (EKAW 2002)*, Sigüenza, Spain, Berlin, Heidelberg.
- Ganitkevitch, J., Van Durme, B., and Callison-Burch, C. 2013. PPDB: the paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'13)*, Atlanta, GA, USA.

- Gil-Lafuente, A. M., and Aluja, J. G. 2012. *Towards an Advanced Modelling of Complex Economic Phenomena: Pretopological and Topological Uncertainty Research Tools*. Berlin, Heidelberg: Springer.
- Glavaš, G., and Ponzetto, S.-P. 2017. Dual tensor model for detecting asymmetric lexico-semantic relations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP'17)*, Copenhagen, Denmark.
- Goikoetxea, J., Sorota, A., and Agirre, E. 2015. Random walks and neural network language models on knowledge bases. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'15)*, Denver, CO, USA.
- Gurevych, I., Eckle-Kohler, J., Hartmann, S., Matuschek, M., Meyer, C. M., and Wirth, C. 2012. UBY – a large-scale unified lexical-semantic resource based on LMF. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL'12)*, Avignon, France.
- Hearst, M. A. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th Conference on Computational Linguistics – Volume 2 (COLING'92)*, Nantes, France.
- Hoffart, J., Suchanek, F. M., Berberich, K., and Weikum, G. 2013. YAGO2: a spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence* **194**: 28–61.
- Hovy, E., Navigli, R., and Ponzetto, S.-P. 2013. Collaboratively built semi-structured content and artificial intelligence: the story so far. *Artificial Intelligence* **194**: 2–27.
- Jauhar, S. K., Dyer, C., and Hovy, E. 2015. Ontologically grounded multi-sense representation learning for semantic vector space models. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'15)*, Denver, CO, USA.
- Jenatton, R., Roux, N. L., Bordes, A., and Obozinski, G. 2012. A latent factor model for highly multi-relational data. In *Proceedings of Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems (NIPS'12)*, Lake Tahoe, NV, USA.
- Jurgens, D., and Pilehvar, M. T. 2016. SemEval-2016 task 14: semantic taxonomy enrichment. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval@NAACL-HLT'16)*, San Diego, CA, USA.
- Kapanipathi, P., Jain, P., Venkataramani, C., and Sheth, A. 2014. User interests identification on Twitter using a hierarchical knowledge base. In *Proceedings of the Semantic Web: Trends and Challenges: 11th International Conference (ESWC'14)*, Cham.
- Klein, D., and Manning, C. D. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics – Volume 1 (ACL'03)*, Sapporo, Japan.
- Klein, P., Ponzetto, S. P., and Glavaš, G. 2017. Improving neural knowledge base completion with cross-lingual projections. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers (EACL'17)*, Valencia, Spain.
- Kozareva, Z., and Hovy, E. 2010. A semi-supervised method to learn and construct taxonomies using the web. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP'10)*, Cambridge, MA, USA.
- Krause, S., Hennig, L., Gabrysak, A., Xu, F., and Uszkoreit, H. 2015. Sar-graphs: a linked linguistic knowledge resource connecting facts with language. In *Proceedings of the 4th Workshop on Linked Data in Linguistics: Resources and Applications*, Beijing, China.
- Lesk, M. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation (SIGDOC'86)*, Toronto, Ontario, Canada.

- Levy, O., and Goldberg, Y. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL'14)*, Baltimore, MD, USA.
- Li, J., and Jurafsky, D. 2015. Do multi-sense embeddings improve natural language understanding? In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*, Lisbon, Portugal.
- Lin, D. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th International Conference on Computational Linguistics – Volume 2 (COLING'98)*, Montreal, Quebec, Canada.
- Lin, T., Pantel, P., Gamon, M., Kannan, A., and Fuxman, A. 2012. Active objects: actions for entity-centric search. In *Proceedings of the 21st International Conference on World Wide Web (WWW'12)*, Lyon, France.
- Maitra, P., and Das, D. 2016. JUNLP at SemEval-2016 Task 13: a language independent approach for hypernym identification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval@NAACL-HLT'16)*, San Diego, CA, USA.
- McCrae, J. P., Fellbaum, C., and Cimiano, P. 2014. Publishing and linking WordNet using lemon and RDF. In *Proceedings of the 3rd Workshop on Linked Data in Linguistics*, Reykjavik, Iceland.
- Mihalcea, R., Chklovski, T., and Kilgarriff, A. 2004. The Senseval-3 English lexical sample task. In *Proceedings of the 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (Senseval-3)*, Barcelona, Spain.
- Mihalcea, R., and Csomai, A. 2007. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM'07)*, Lisbon, Portugal.
- Mihalcea, R., and Moldovan, D. I. 2001. eXtended WordNet: progress report. In *Proceedings of NAACL Workshop on WordNet and Other Lexical Resources*, Pittsburgh, PA, USA.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems (NIPS'13)*, Lake Tahoe, NV, USA.
- Mikolov, T., Yih, W.-T., and Zweig, G. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'13)*, Atlanta, GA, USA.
- Miller, G. A., Leacock, C., Tengi, R., and Bunker, R. T. 1993. A semantic concordance. In *Proceedings of the Workshop on Human Language Technology*, Princeton, NJ, USA.
- Mintz, M., Bills, S., Snow, R., and Jurafsky, D. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Vol. 2 (ACL'09)*, Suntec, Singapore.
- Nastase, V., and Strube, M. 2013. Transforming Wikipedia into a large scale multilingual concept network. *Artificial Intelligence* **194**: 62–85.
- Navigli, R. 2009. Word sense disambiguation: a survey. *ACM CSUR* **41**(2): 1–69.
- Navigli, R., and Ponzetto, S. P. 2012a. BabelNet: the automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence* **193**: 217–50.
- Navigli, R., and Ponzetto, S. P. 2012b. Joining forces pays off: multilingual joint Word Sense Disambiguation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL'12)*, Jeju Island, South Korea.
- Navigli, R., and Velardi, P. 2010. Learning Word-class lattices for definition and hypernym extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL'10)*, Uppsala, Sweden.

- Neelakantan, A., Shankar, J., Passos, A., and McCallum A. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*, Doha, Qatar.
- Nickel, M., Murphy, K., Tresp, V., and Gabrilovich, E. 2016. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE* **104**(1): 11–33.
- Niemann, E., and Gurevych, I. 2011. The people's web meets linguistic knowledge: automatic sense alignment of Wikipedia and wordnet. In *Proceedings of the Ninth International Conference on Computational Semantics (IWCS'11)*, Oxford, UK.
- Nieto Piña, L., and Johansson, R. 2016. Embedding senses for efficient graph-based word sense disambiguation. In *Proceedings of the 2016 Workshop on Graph-based Methods for Natural Language Processing (Textgraphs'16)*, San Diego, CA, USA.
- Norvig, P. 2016. The semantic web and the semantics of the web: where does meaning come from? In *Proceedings of the 25th International Conference on World Wide Web (WWW'16)*, Montreal, Quebec, Canada.
- Padó, S., and Lapata M. 2007. Dependency-based construction of semantic space models. *Computational Linguistics* **33**(2): 161–99.
- Panchenko, A. 2016. Best of both worlds: making word sense embeddings interpretable. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC'16)*, Portorož, Slovenia.
- Panchenko, A., Faralli, S., Ponzetto, S. P., and Biemann, C. 2017a. Using linked disambiguated distributional networks for word sense disambiguation. In *Proceedings of the 1st EACL Workshop on Sense, Concept and Entity Representations and their Applications*, Valencia, Spain.
- Panchenko, A., Faralli, S., Ruppert, E., Remus, S., Naets, H., Fairon, C., Ponzetto, S. P., and Biemann, C. 2016. TAXI at SemEval-2016 task 13: a taxonomy induction method based on lexico-syntactic patterns, substrings and focused crawling. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval@NAACL-HLT'16)*, San Diego, CA, USA.
- Panchenko, A., Marten, F., Ruppert, E., Faralli, S., Ustalov, D., Ponzetto, S. P., and Biemann, C. 2017b. Unsupervised, knowledge-free, and interpretable word sense disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP'17)*, Copenhagen, Denmark.
- Panchenko, A., and Morozova, O. 2012. A study of hybrid similarity measures for semantic relation extraction. In *Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data*, Avignon, France.
- Panchenko, A., Romanov, P., Morozova, O., Naets, H., Philippovich, A., Romanov, A., and Fairon, C. 2013. Serelex: search and visualization of semantically related words. In *Proceedings of the European Conference on Information Retrieval (ECIR'13)*, Moscow, Russia.
- Panchenko, A., Ruppert, E., Faralli, S., Ponzetto, S. P., and Biemann C. 2017c. Unsupervised does not mean uninterpretable: the case for word sense induction and disambiguation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL'17)*, Valencia, Spain.
- Panchenko, A., Simon, J., Riedl, M., and Biemann, C. 2016. Noun sense induction and disambiguation using graph-based distributional semantics. In *Proceedings of the 13th Conference on Natural Language Processing (KONVENS'16)*, Bochum, Germany.
- Parker, R., Graff, D., Kong, J., Chen, K., and Maeda, K. 2011. *English Gigaword*, 5th ed. Philadelphia: Linguistic Data Consortium.
- Pavel, S., and Euzenat, J. 2013. Ontology matching: state of the art and future challenges. *IEEE Transaction on Knowledge and Data Engineering* **25**(1): 158–76.

- Pedersen, T., Patwardhan, S., and Michelizzi, J. 2004. WordNet::similarity – measuring the relatedness of concepts. In *Proceedings of the HLT-NAACL 2004: Demonstration Papers (HLT-NAACL'04 Demos)*, Boston, MA, USA.
- Pelevina, M., Arefiev, N., Biemann, C., and Panchenko, A. 2016. Making sense of word embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, Berlin, Germany.
- Pennington, J., Socher, R., and Manning, C. 2014. GloVe: global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*, Doha, Qatar.
- Pham, N. T., Lazaridou, A., and Baroni, M. 2015. A multitask objective to inject lexical contrast into distributional semantics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Volume 2: Short Papers (ACL'15)*, Beijing, China.
- Pocostales, J. 2016. NUIG-UNLP at SemEval-2016 task 13: a simple word embedding-based approach for taxonomy extraction. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval@NAACL-HLT'16)*, San Diego, CA, USA.
- Ponzetto, S. P., and Navigli, R. 2009. Large-scale taxonomy mapping for restructuring and integrating Wikipedia. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09)*, Pasadena, CA, USA.
- Ponzetto, S. P., and Navigli, R. 2010. Knowledge-rich word sense disambiguation rivaling supervised systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL'10)*, Uppsala, Sweden.
- Ponzetto, S. P., and Strube, M. 2011. Taxonomy induction based on a collaboratively built knowledge repository. *Artificial Intelligence* **175**: 1737–56.
- Pradhan, S. S., Loper, E., Dligach, D., and Palmer, M. 2007. SemEval-2007 task 17: english lexical sample, SRL and all words. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval'07)*, Prague, Czech Republic.
- Reisinger, J., and Mooney, R. J. 2010. Multi-prototype vector-space models of word meaning. In *Proceedings of the Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT'10)*, Los Angeles, CA, USA.
- Richter, M., Quasthoff, U., Hallsteinsdóttir, E., and Biemann, C. 2006. Exploiting the Leipzig corpora collection. In *Proceedings of IS-LTC'06*, Ljubljana, Slovenia.
- Riedl, M. 2016. *Unsupervised Methods for Learning and Using Semantics of Natural Language*. Ph.D. thesis. Germany: TU Darmstadt. <http://tuprints.ulb.tu-darmstadt.de/5435/>.
- Riedl, M., and Biemann, C. 2013. Scaling to large³ data: an efficient and effective method to compute distributional thesauri. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, a Meeting of SIGDAT, a Special Interest Group of the ACL (EMNLP'13)*, Seattle, WA, USA.
- Riedl, M., and Biemann, C. 2015. A single word is not enough: ranking multiword expressions using distributional semantics. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*, Lisbon, Portugal.
- Roller, S., Erk, K., and Boleda, G. 2014. Inclusive yet selective: supervised distributional hypernymy detection. In *Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers (COLING'14)*, Dublin, Ireland.
- Rospocher, M., van Erp, M., Vossen, P., Fokkens, A., Aldabe, I., Rigau, G., Soroa, A., Ploeger, T., and Bogaard, T. 2016. Building event-centric knowledge graphs from news. *Journal of Web Semantics* **37–38**: 132–51.
- Rothe, S., and Schütze, H. 2015. AutoExtend: extending word embeddings to embeddings for synsets and lexemes. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language*

- Processing of the Asian Federation of Natural Language Processing, Volume 1: Long Papers (ACL'15)*, Beijing, China.
- Ruppert, E., Kaufmann, M., Riedl, M., and Biemann, C. 2015. JoBimViz: a web-based visualization for graph-based distributional semantic models. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, System Demonstrations (ACL'15)*, Beijing, China.
- Schuhmacher, M., Dietz, L., and Ponzetto, S. P. 2015. Ranking entities for web queries through text and knowledge. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM'15)*, Melbourne, Australia.
- Shwartz, V., Goldberg, Y., and Dagan, I. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers (ACL'16)*, Berlin, Germany.
- Snow, R., Jurafsky, D., and Ng, A. 2006. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (ACL'06)*, Sydney, Australia.
- Snow, R., Jurafsky, D., and Ng, A. Y. 2004. Learning syntactic patterns for automatic hypernym discovery. In *Proceedings of the Advances in Neural Information Processing Systems 17 (NIPS'04)*, Vancouver, BC, Canada.
- Socher, R., Chen, D., Manning, C. D., and Ng, A. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of the Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems (NIPS'13)*, Lake Tahoe, NV, USA.
- Suchanek, F. M., Kasneci, G., and Weikum, G. 2008. YAGO: a large ontology from Wikipedia and WordNet. *Journal of Web Semantics* **6**(3): 203–17.
- Tan, L., Bond, F., and van Genabith, J. 2016. USAAR at SemEval-2016 task 13: hyponym endocentricity. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval@NAACL-HLT'16)*, San Diego, CA, USA.
- Tarjan, R. 1972. Depth-first search and linear graph algorithms. *SIAM Journal on Computing* **1**(2): 146–60.
- Turney, P. D., and Pantel, P. 2010. From frequency to meaning: vector space models of semantics. *Journal of Artificial Intelligence Research* **37**: 141–88.
- Van de Cruys, T. 2010. Mining for meaning: the extraction of lexico-semantic knowledge from text. *Groningen Dissertations in Linguistics* **82**.
- Van Dongen, S. 2008. Graph clustering via a discrete uncoupling process. *SIAM Journal on Matrix Analysis and Applications* **30**(1): 121–41.
- Velardi, P., Faralli, S., and Navigli, R. 2013. OntoLearn reloaded: a graph-based algorithm for taxonomy induction. *Computational Linguistics* **39**(3): 665–707.
- Velardi, P., Navigli, R., Faralli, S., and Ruiz-Martínez, J. M. 2012. A new method for evaluating automatically learned terminological taxonomies. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey.
- Véronis, J. 2004. HyperLex: lexical cartography for information retrieval. *Computer Speech and Language* **18**: 223–52.
- Wang, Z., Li, J., Wang, Z., and Tang, J. 2012. Cross-lingual knowledge linking across Wiki knowledge bases. In *Proceedings of the 21st International Conference on World Wide Web (WWW'12)*, Lyon, France.
- Weeds, J., Weir, D., and McCarthy, D. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING'04)*, Geneva, Switzerland.

- West, R., Gabrilovich, E., Murphy, K., Sun, S., Gupta, R., and Lin, D. 2014. Knowledge base completion via search-based question answering. In *Proceedings of the 23rd International Conference on World Wide Web (WWW'14)*, Seoul, South Korea.
- Wu, W., Li, H., Wang, H., and Zhu, K. Q. 2012. Probase: a probabilistic taxonomy for text understanding. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data (SIGMOD'12)*, Scottsdale, AZ, USA.
- Yu, M., and Dredze, M. 2014. Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Volume 2: Short Papers (ACL'14)*, Baltimore, MD, USA.

A.9 Paper “Watset: Automatic Induction of Synsets from a Graph of Synonyms”

D. Ustalov, [A. Panchenko](#), and C. Biemann, “Watset: Automatic Induction of Synsets from a Graph of Synonyms,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Vancouver, Canada), pp. 1579–1590, Association for Computational Linguistics, July 2017

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/P17-1145>

Watset: Automatic Induction of Synsets from a Graph of Synonyms

Dmitry Ustalov^{†*}, Alexander Panchenko[‡], and Chris Biemann[‡]

[†]Institute of Natural Sciences and Mathematics, Ural Federal University, Russia

^{*}Krasovskii Institute of Mathematics and Mechanics, Russia

[‡]Language Technology Group, Department of Informatics, Universität Hamburg, Germany

dmitry.ustalov@urfu.ru

{panchenko, biemann}@informatik.uni-hamburg.de

Abstract

This paper presents a new graph-based approach that induces synsets using synonymy dictionaries and word embeddings. First, we build a weighted graph of synonyms extracted from commonly available resources, such as Wiktionary. Second, we apply word sense induction to deal with ambiguous words. Finally, we cluster the disambiguated version of the ambiguous input graph into synsets. Our meta-clustering approach lets us use an efficient hard clustering algorithm to perform a fuzzy clustering of the graph. Despite its simplicity, our approach shows excellent results, outperforming five competitive state-of-the-art methods in terms of F-score on three gold standard datasets for English and Russian derived from large-scale manually constructed lexical resources.

1 Introduction

A *synset* is a set of mutual synonyms, which can be represented as a clique graph where nodes are words and edges are synonymy relations. Synsets represent word senses and are building blocks of WordNet (Miller, 1995) and similar resources such as thesauri and lexical ontologies. These resources are crucial for many natural language processing applications that require common sense reasoning, such as information retrieval (Gong et al., 2005) and question answering (Kwok et al., 2001; Zhou et al., 2013). However, for most languages, no manually-constructed resource is available that is comparable to the English WordNet in terms of coverage and quality. For instance, Kiselev et al. (2015) present a comparative analysis of lexical resources available for the Russian

language concluding that there is no resource compared to WordNet in terms of coverage and quality for Russian. This lack of linguistic resources for many languages urges the development of new methods for automatic construction of WordNet-like resources. The automatic methods foster construction and use of the new lexical resources.

Wikipedia¹, Wiktionary², OmegaWiki³ and other collaboratively-created resources contain a large amount of lexical semantic information—yet designed to be human-readable and not formally structured. While semantic relations can be automatically extracted using tools such as DKPro JWKT⁴ and Wikokit⁵, words in these relations are not disambiguated. For instance, the synonymy pairs (*bank*, *streambank*) and (*bank*, *banking company*) will be connected via the word “bank”, while they refer to the different senses. This problem stems from the fact that articles in Wiktionary and similar resources list undisambiguated synonyms. They are easy to disambiguate for humans while reading a dictionary article, but can be a source of errors for language processing systems.

The contribution of this paper is a novel approach that resolves ambiguities in the input graph to perform fuzzy clustering. The method takes as an input synonymy relations between potentially ambiguous terms available in human-readable dictionaries and transforms them into a machine readable representation in the form of disambiguated synsets. Our method, called WATSET, is based on a new local-global meta-algorithm for fuzzy graph clustering. The underlying principle is to discover the word senses based on a *local* graph cluster-

¹<http://www.wikipedia.org>

²<http://www.wiktionary.org>

³<http://www.omegawiki.org>

⁴<https://dkpro.github.io/dkpro-jwktl>

⁵<https://github.com/componavt/wikokit>

ing, and then to induce synsets using *global* sense clustering. We show that our method outperforms other methods for synset induction. The induced resource eliminates the need in manual synset construction and can be used to build WordNet-like semantic networks for under-resourced languages. An implementation of our method along with induced lexical resources is available online.⁶

2 Related Work

Methods based on resource linking surveyed by Gurevych et al. (2016) gather various existing lexical resources and perform their linking to obtain a machine-readable repository of lexical semantic knowledge. For instance, BabelNet (Navigli and Ponzetto, 2012) relies in its core on a linking of WordNet and Wikipedia. UBY (Gurevych et al., 2012) is a general-purpose specification for the representation of lexical-semantic resources and links between them. The main advantage of our approach compared to the lexical resources is that no manual synset encoding is required.

Methods based on word sense induction try to induce sense representations without the need for any initial lexical resource by extracting semantic relations from text. In particular, word sense induction (WSI) based on word ego networks clusters graphs of semantically related words (Lin, 1998; Pantel and Lin, 2002; Dorow and Widdows, 2003; Véronis, 2004; Hope and Keller, 2013; Pelevina et al., 2016; Panchenko et al., 2017a), where each cluster corresponds to a word sense. An ego network consists of a single node (ego) together with the nodes they are connected to (alters) and all the edges among those alters (Everett and Borgatti, 2005). In the case of WSI, such a network is a local neighborhood of one word. Nodes of the ego network are the words which are semantically similar to the target word.

Such approaches are able to discover homonymous senses of words, e.g., “bank” as slope versus “bank” as organisation (Di Marco and Navigli, 2012). However, as the graphs are usually composed of semantically related words obtained using distributional methods (Baroni and Lenci, 2010; Biemann and Riedl, 2013), the resulting clusters by no means can be considered synsets. Namely, (1) they contain words related not only via synonymy relation, but via a mixture of relations such as synonymy, hypernymy,

co-hyponymy, antonymy, etc. (Heylen et al., 2008; Panchenko, 2011); (2) clusters are not unique, i.e., one word can occur in clusters of different ego networks referring to the same sense, while in WordNet a word sense occurs only in a single synset.

In our synset induction method, we use word ego network clustering similarly as in word sense induction approaches, but apply them to a graph of semantically clean synonyms.

Methods based on clustering of synonyms, such as our approach, induce the resource from an ambiguous graph of synonyms where edges are extracted from manually-created resources. According to the best of our knowledge, most experiments either employed graph-based word sense induction applied to text-derived graphs or relied on a linking-based method that already assumes availability of a WordNet-like resource. A notable exception is the ECO approach by Gonçalo Oliveira and Gomes (2014), which was applied to induce a WordNet of the Portuguese language called Onto.PT.⁷ We compare to this approach and to five other state-of-the-art graph clustering algorithms as the baselines.

ECO (Gonçalo Oliveira and Gomes, 2014) is a fuzzy clustering algorithm that was used to induce synsets for a Portuguese WordNet from several available synonymy dictionaries. The algorithm starts by adding random noise to edge weights. Then, the approach applies Markov Clustering (see below) of this graph several times to estimate the probability of each word pair being in the same synset. Finally, candidate pairs over a certain threshold are added to output synsets.

MaxMax (Hope and Keller, 2013) is a fuzzy clustering algorithm particularly designed for the word sense induction task. In a nutshell, pairs of nodes are grouped if they have a maximal mutual affinity. The algorithm starts by converting the undirected input graph into a directed graph by keeping the maximal affinity nodes of each node. Next, all nodes are marked as root nodes. Finally, for each root node, the following procedure is repeated: all transitive children of this root form a cluster and the root are marked as non-root nodes; a root node together with all its transitive children form a fuzzy cluster.

Markov Clustering (MCL) (van Dongen, 2000) is a hard clustering algorithm for graphs based on simulation of stochastic flow in graphs.

⁶<https://github.com/dustalov/watset>

⁷<http://ontopt.dei.uc.pt>

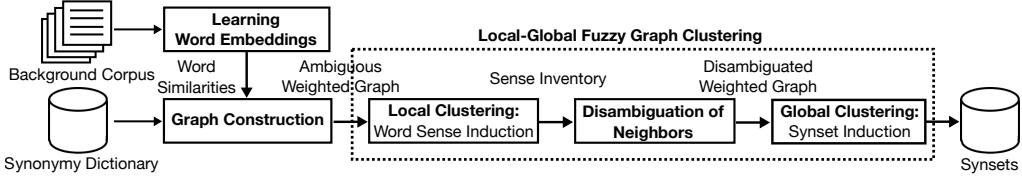


Figure 1: Outline of the WATSET method for synset induction.

MCL simulates random walks within a graph by alternation of two operators called expansion and inflation, which recompute the class labels. Notably, it has been successfully used for the word sense induction task (Dorow and Widdows, 2003).

Chinese Whispers (CW) (Biemann, 2006) is a *hard* clustering algorithm for weighted graphs that can be considered as a special case of MCL with a simplified class update step. At each iteration, the labels of all the nodes are updated according to the majority labels among the neighboring nodes. The algorithm has a meta-parameter that controls graph weights that can be set to three values: (1) *top* sums over the neighborhood’s classes; (2) *nolog* downgrades the influence of a neighboring node by its degree or by (3) *log* of its degree.

Clique Percolation Method (CPM) (Palla et al., 2005) is a *fuzzy* clustering algorithm for unweighted graphs that builds up clusters from k -cliques corresponding to fully connected subgraphs of k nodes. While this method is only commonly used in social network analysis, we decided to add it to the comparison as synsets are essentially cliques of synonyms, which makes it natural to apply an algorithm based on clique detection.

3 The WATSET Method

The goal of our method is to induce a set of unambiguous synsets by grouping individual ambiguous synonyms. An outline of the proposed approach is depicted in Figure 1. The method takes a dictionary of ambiguous synonymy relations and a text corpus as an input and outputs synsets. Note that the method can be used without a background corpus, yet as our experiments will show, corpus-based information improves the results when utilizing it for weighting the word graph’s edges.

A synonymy dictionary can be perceived as a graph, where the nodes correspond to lexical entries (words) and the edges connect pairs of the nodes when the synonymy relation between them holds. The cliques in such a graph naturally form

densely connected sets of synonyms corresponding to concepts (Gfeller et al., 2005). Given the fact that solving the clique problem exactly in a graph is NP-complete (Bomze et al., 1999) and that these graphs typically contain tens of thousands of nodes, it is reasonable to use efficient hard graph clustering algorithms, like MCL and CW, for finding a global segmentation of the graph. However, the hard clustering property of these algorithm does not handle polysemy: while one word could have several senses, it will be assigned to only one cluster. To deal with this limitation, a word sense induction procedure is used to induce senses for all words, one at the time, to produce a disambiguated version of the graph where a word is now represented with one or many word senses. The concept of a disambiguated graph is described in (Biemann, 2012). Finally, the disambiguated word sense graph is clustered globally to induce synsets, which are hard clusters of word senses.

More specifically, the method consists of five steps presented in Figure 1: (1) learning word embeddings; (2) constructing the ambiguous weighted graph of synonyms G ; (3) inducing the word senses; (4) constructing the disambiguated weighted graph G' by disambiguating of neighbors with respect to the induced word senses; (5) global clustering of the graph G' .

3.1 Learning Word Embeddings

Since different graph clustering algorithms are sensitive to edge weighting, we consider distributional semantic similarity based on word embeddings as a possible edge weighting approach for our synonymy graph. As we show further, this approach improves over unweighted versions and yields the best overall results.

3.2 Construction of a Synonymy Graph

We construct the synonymy graph $G = (V, E)$ as follows. The set of nodes V includes every lexeme appearing in the input synonymy dictionaries. The set of undirected edges E is composed of all edges

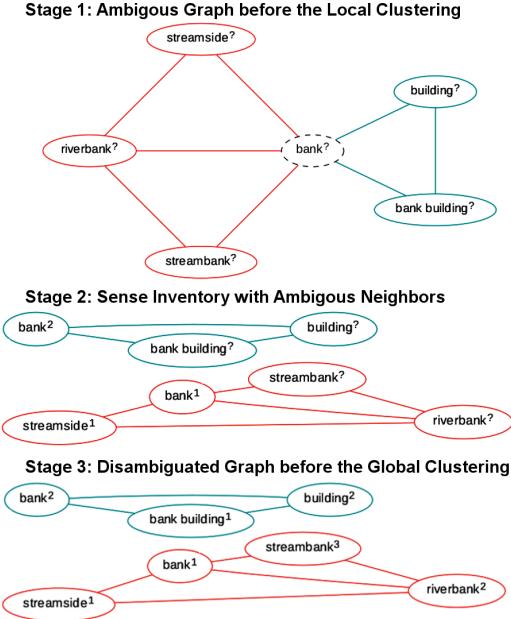


Figure 2: Disambiguation of an ambiguous input graph using local clustering (WSI) to facilitate global clustering of words into synsets.

$(u, v) \in V \times V$ retrieved from one of the input synonymy dictionaries. We consider three edge weight representations:

- **ones** that assigns every edge the constant weight of 1;
- **count** that weights the edge (u, v) as the number of times the synonymy pair appeared in the input dictionaries;
- **sim** that assigns every edge (u, v) a weight equal to the cosine similarity of skip-gram word vectors (Mikolov et al., 2013).

As the graph G is likely to have polysemous words, the goal is to separate individual word senses using graph-based word sense induction.

3.3 Local Clustering: Word Sense Induction

In order to facilitate global fuzzy clustering of the graph, we perform disambiguation of its ambiguous nodes as illustrated in Figure 2. First, we use a graph-based word sense induction method that is similar to the curvature-based approach of Dorow and Widdows (2003). In particular, removal of the nodes participating in many triangles tends to

separate the original graph into several connected components. Thus, given a word u , we extract a network of its nearest neighbors from the synonymy graph G . Then, we remove the original word u from this network and run a hard graph clustering algorithm that assigns one node to one and only one cluster. In our experiments, we test Chinese Whispers and Markov Clustering. The expected result of this is that each cluster represents a different sense of the word u , e.g.:

$$\begin{aligned} \text{bank}^1 &= \{\text{streambank}, \text{riverbank}, \dots\} \\ \text{bank}^2 &= \{\text{bank company}, \dots\} \\ \text{bank}^3 &= \{\text{bank building}, \text{building}, \dots\} \\ \text{bank}^4 &= \{\text{coin bank}, \text{penny bank}, \dots\} \end{aligned}$$

We denote, e.g., bank^1 , bank^2 and other items as word senses referred to as $\text{senses}(\text{bank})$. We denote as $\text{ctx}(s)$ a cluster corresponding to the word sense s . Note that the context words have no sense labels. They are recovered by the disambiguation approach described next.

3.4 Disambiguation of Neighbors

Next, we disambiguate the neighbors of each induced sense. The previous step results in splitting word nodes into (one or more) sense nodes. However, nearest neighbors of each sense node are still ambiguous, e.g., $(\text{bank}^3, \text{building}?)$. To recover these sense labels of the neighboring words, we employ the following sense disambiguation approach proposed by Faralli et al. (2016). For each word u in the context $\text{ctx}(s)$ of the sense s , we find the most similar sense of that word \hat{u} to the context. We use the cosine similarity measure between the context of the sense s and the context of each candidate sense u' of the word u :

$$\hat{u} = \arg \max_{u' \in \text{senses}(u)} \cos(\text{ctx}(s), \text{ctx}(u')).$$

A context $\text{ctx}(\cdot)$ is represented by a sparse vector in a vector space of all ambiguous words of all contexts. The result is a disambiguated context $\widehat{\text{ctx}}(s)$ in a space of *disambiguated* words derived from its *ambiguous* version $\text{ctx}(s)$:

$$\widehat{\text{ctx}}(s) = \{\hat{u} : u \in \text{ctx}(s)\}.$$

3.5 Global Clustering: Synset Induction

Finally, we construct the word sense graph $G' = (V', E')$ using the disambiguated senses instead of the original words and establishing the edges between these disambiguated senses:

$$V' = \bigcup_{u \in V} \text{senses}(u), \quad E' = \bigcup_{s \in V'} \{s\} \times \widehat{\text{ctx}}(s).$$

Running a hard clustering algorithm on G' produces the desired set of synsets as our final result. Figure 2 illustrates the process of disambiguation of an input ambiguous graph on the example of the word ‘‘bank’’. As one may observe, disambiguation of the nearest neighbors is a necessity to be able to construct a global version of the sense-aware graph. Note that current approaches to WSI, e.g., (Véronis, 2004; Biemann, 2006; Hope and Keller, 2013), do not perform this step, but perform only local clustering of the graph since they do not aim at a global representation of synsets.

3.6 Local-Global Fuzzy Graph Clustering

While we use our approach to synset induction in this work, the core of our method is the ‘‘local-global’’ fuzzy graph clustering algorithm, which can be applied to arbitrary graphs (see Figure 1). This method, summarized in Algorithm 1, takes an undirected graph $G = (V, E)$ as the input and outputs a set of fuzzy clusters of its nodes V . This is a meta-algorithm as it operates on top of two hard clustering algorithms denoted as $\text{Cluster}_{\text{local}}$ and $\text{Cluster}_{\text{global}}$, such as CW or MCL. At the first phase of the algorithm, for each node its senses are induced via ego network clustering (lines 1–7). Next, the disambiguation of each ego network is performed (lines 8–15). Finally, the fuzzy clusters are obtained by applying the hard clustering algorithm to the disambiguated graph (line 16). As a post-processing step, the sense labels can be removed to make the cluster elements subsets of V .

4 Evaluation

We conduct our experiments on resources from two different languages. We evaluate our approach on two datasets for English to demonstrate its performance on a resource-rich language. Additionally, we evaluate it on two Russian datasets since Russian is a good example of an under-resourced language with a clear need for synset induction.

4.1 Gold Standard Datasets

For each language, we used two differently constructed lexical semantic resources listed in Table 1 to obtain gold standard synsets.

English. We use **WordNet**⁸, a popular English lexical database constructed by expert lexicographers. WordNet contains general vocabulary and

Algorithm 1 WATSET fuzzy graph clustering

Input: a set of nodes V and a set of edges E .
Output: a set of fuzzy clusters of V .

```

1: for all  $u \in V$  do
2:    $C \leftarrow \text{Cluster}_{\text{local}}(\text{Ego}(u)) // C = \{C_1, \dots\}$ 
3:   for  $i \leftarrow 1 \dots |C|$  do
4:      $\text{ctx}(u^i) \leftarrow C_i$ 
5:      $\text{senses}(u) \leftarrow \text{senses}(u) \cup \{u^i\}$ 
6:   end for
7: end for
8:  $V' \leftarrow \bigcup_{u \in V} \text{senses}(u)$ 
9: for all  $s \in V'$  do
10:   for all  $u \in \text{ctx}(s)$  do
11:      $\hat{u} \leftarrow \arg \max_{u' \in \text{senses}(u)} \cos(\text{ctx}(s), \text{ctx}(u'))$ 
12:   end for
13:    $\widehat{\text{ctx}}(s) \leftarrow \{\hat{u} : u \in \text{ctx}(s)\}$ 
14: end for
15:  $E' \leftarrow \bigcup_{s \in V'} \{s\} \times \widehat{\text{ctx}}(s)$ 
16: return  $\text{Cluster}_{\text{global}}(V', E')$ 
```

appears to be *de facto* gold standard in similar tasks (Hope and Keller, 2013). We used WordNet 3.1 to derive the synonymy pairs from synsets. Additionally, we use **BabelNet**⁹, a large-scale multilingual semantic network constructed automatically using WordNet, Wikipedia and other resources. We retrieved all the synonymy pairs from the BabelNet 3.7 synsets marked as English.

Russian. As a lexical ontology for Russian, we use **RuWordNet**¹⁰ (Loukachevitch et al., 2016), containing both general vocabulary and domain-specific synsets related to sport, finance, economics, etc. Up to a half of the words in this resource are multi-word expressions (Kiselev et al., 2015), which is due to the coverage of domain-specific vocabulary. RuWordNet is a WordNet-like version of the RuThes thesaurus that is constructed in the traditional way, namely by a small group of expert lexicographers (Loukachevitch, 2011). In addition, we use **Yet Another Russian WordNet**¹¹ (YARN) by Braslavski et al. (2016) as another gold standard for Russian. The resource is constructed using crowdsourcing and mostly covers general vocabulary. Particularly, non-expert users are allowed to edit synsets in a collaborative way loosely supervised by a team of project curators. Due to the ongoing development of the re-

⁹<http://www.babelnet.org>

¹⁰<http://ruwordnet.ru/en>

¹¹<https://russianword.net/en>

⁸<https://wordnet.princeton.edu>

source, we selected as the gold standard only those synsets that were edited at least eight times in order to filter out noisy incomplete synsets.

Resource		# words	# synsets	# synonyms
WordNet	En	148 730	117 659	152 254
BabelNet	En	11 710 137	6 667 855	28 822 400
RuWordNet	Ru	110 242	49 492	278 381
YARN	Ru	9 141	2 210	48 291

Table 1: Statistics of the gold standard datasets.

4.2 Evaluation Metrics

To evaluate the quality of the induced synsets, we transformed them into binary synonymy relations and computed precision, recall, and F-score on the basis of the overlap of these binary relations with the binary relations from the gold standard datasets. Given a synset containing n words, we generate a set of $\frac{n(n-1)}{2}$ pairs of synonyms. The F-score calculated this way is known as *Paired F-score* (Manandhar et al., 2010; Hope and Keller, 2013). The advantage of this measure compared to other cluster evaluation measures, such as *Fuzzy B-Cubed* (Jurgens and Klapaftis, 2013), is its straightforward interpretability.

4.3 Word Embeddings

English. We use the standard 300-dimensional word embeddings trained on the 100 billion tokens Google News corpus (Mikolov et al., 2013).¹²

Russian. We use the 500-dimensional word embeddings trained using the skip-gram model with negative sampling (Mikolov et al., 2013) using a context window size of 10 with the minimal word frequency of 5 on a 12.9 billion tokens corpus of books. These embeddings were shown to produce state-of-the-art results in the RUSSE shared task¹³ and are part of the Russian Distributional Thesaurus (RDT) (Panchenko et al., 2017b).¹⁴

4.4 Input Dictionary of Synonyms

For each language, we constructed a synonymy graph using openly available language resources. The statistics of the graphs used as the input in the further experiments are shown in Table 2.

¹²<https://code.google.com/p/word2vec>

¹³http://www.dialog-21.ru/en/evaluation/2015/semantic_similarity

¹⁴<http://russe.nlp.ru/downloads>

English. Synonyms were extracted from the English Wiktionary¹⁵, which is the largest Wiktionary at the present moment in terms of the lexical coverage, using the DKPro JWTKL tool by Zesch et al. (2008). English words have been extracted from the dump.

Russian. Synonyms from three sources were combined to improve lexical coverage of the input dictionary and to enforce confidence in jointly observed synonyms: (1) synonyms listed in the Russian Wiktionary extracted using the Wikokit tool by Krizhanovsky and Smirnov (2013); (2) the dictionary of Abramov (1999); and (3) the Universal Dictionary of Concepts (Dikonov, 2013). While the two latter resources are specific to Russian, Wiktionary is available for most languages. Note that the same input synonymy dictionaries were used by authors of YARN to construct synsets using crowdsourcing. The results on the YARN dataset show how close an automatic synset induction method can approximate manually created synsets provided the same starting material.¹⁶

Language	# words	# synonyms
English	243 840	212 163
Russian	83 092	211 986

Table 2: Statistics of the input datasets.

5 Results

We compare WATSET with five state-of-the art graph clustering methods presented in Section 2: Chinese Whispers (CW), Markov Clustering (MCL), MaxMax, ECO clustering, and the clique percolation method (CPM). The first two algorithms perform hard clustering, while the last three are fuzzy clustering methods just like our method. While the hard clustering algorithms are able to discover clusters which correspond to synsets composed of unambiguous words, they can produce wrong results in the presence of lexical ambiguity (one node belongs to several synsets). In our experiments, we rely on our own implementation of MaxMax and ECO as reference implementations are not available. For CW¹⁷, MCL¹⁸

¹⁵We used the Wiktionary dumps of February 1, 2017.

¹⁶We used the YARN dumps of February 7, 2017.

¹⁷<https://www.github.com/uhh-1t/chinese-whispers>

¹⁸<http://java-ml.sourceforge.net>

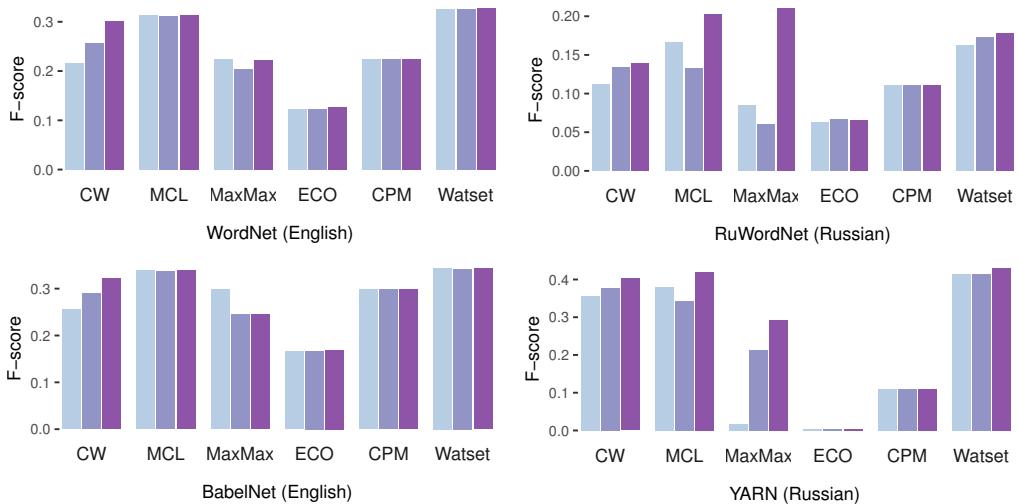


Figure 3: Impact of the different graph weighting schemas on the performance of synset induction: █ ones, █ count, █ sim. Each bar corresponds to the top performance of a method in Tables 3 and 4.

and CPM¹⁹, available implementations have been used. During the evaluation, we delete clusters equal or larger than the threshold of 150 words as they hardly can represent any meaningful synset. The notation WATSET[MCL, CW_{top}] means using MCL for local clustering and Chinese Whispers in the *top* mode for global clustering.

5.1 Impact of Graph Weighting Schema

Figure 3 presents an overview of the evaluation results on both datasets. The first step, common for all of the tested synset induction methods, is graph construction. Thus, we started with an analysis of three ways to weight edges of the graph introduced in Section 3.2: binary scores (*ones*), frequencies (*count*), and semantic similarity scores (*sim*) based on word vector similarity. Results across various configurations and methods indicate that using the weights based on the similarity scores provided by word embeddings is the best strategy for all methods except MaxMax on the English datasets. However, its performance using the *ones* weighting does not exceed the other methods using the *sim* weighting. Therefore, we report all further results on the basis of the *sim* weights. The edge weighting scheme impacts Russian more for most algorithms. The CW algorithm however remains sensitive to the weighting also for the English dataset due to its randomized nature.

5.2 Comparative Analysis

Table 3 and 4 present evaluation results for both languages. For each method, we show the best configurations in terms of F-score. One may note that the granularity of the resulting synsets, especially for Russian, is very different, ranging from 4 000 synsets for the CPM_{k=3} method to 67 645 induced by the ECO method. Both tables report the number of words, synsets and synonyms after pruning huge clusters larger than 150 words. Without this pruning, the MaxMax and CPM methods tend to discover giant components obtaining almost zero precision as we generate all possible pairs of nodes in such clusters. The other methods did not show such behavior.

WATSET robustly outperforms all other methods according to F-score on both English datasets (Table 3) and on the YARN dataset for Russian (Table 4). Also, it outperforms all other methods according to recall on both Russian datasets. The disambiguation of the input graph performed by the WATSET method splits nodes belonging to several local communities to several nodes, significantly facilitating the clustering task otherwise complicated by the presence of the hubs that wrongly link semantically unrelated nodes.

Interestingly, in all the cases, the toughest competitor was a hard clustering algorithm—MCL (van Dongen, 2000). We observed that the “plain” MCL successfully groups monosemous words, but

¹⁹<https://networkx.github.io>

Method	# words	# synsets	# synonyms	WordNet			BabelNet		
				P	R	F1	P	R	F1
WATSET[MCL, MCL]	243 840	112 267	345 883	0.345	0.308	0.325	0.400	0.301	0.343
MCL	243 840	84 679	387 315	0.342	0.291	0.314	0.390	0.300	0.339
WATSET[MCL, CW _{log}]	243 840	105 631	431 085	0.314	0.325	0.319	0.359	0.312	0.334
CW _{top}	243 840	77 879	539 753	0.285	0.317	0.300	0.326	0.317	0.321
WATSET[CW _{log} , MCL]	243 840	164 689	227 906	0.394	0.280	0.327	0.439	0.245	0.314
WATSET[CW _{log} , CW _{log}]	243 840	164 667	228 523	0.392	0.280	0.327	0.439	0.245	0.314
CPM _{k=2}	186 896	67 109	317 293	0.561	0.141	0.225	0.492	0.214	0.299
MaxMax	219 892	73 929	797 743	0.176	0.300	0.222	0.202	0.313	0.245
ECO	243 840	171 773	84 372	0.784	0.069	0.128	0.699	0.096	0.169

Table 3: Comparison of the synset induction methods on datasets for English. All methods rely on the similarity edge weighting (*sim*); best configurations of each method in terms of F-scores are shown for each dataset. Results are sorted by F-score on BabelNet, top three values of each metric are boldfaced.

Method	# words	# synsets	# synonyms	RuWordNet			YARN		
				P	R	F1	P	R	F1
WATSET[CW _{nolog} , MCL]	83 092	55 369	332 727	0.120	0.349	0.178	0.402	0.463	0.430
WATSET[MCL, MCL]	83 092	36 217	403 068	0.111	0.341	0.168	0.405	0.455	0.428
WATSET[CW _{top} , CW _{log}]	83 092	55 319	341 043	0.116	0.351	0.174	0.386	0.474	0.425
MCL	83 092	21 973	353 848	0.155	0.291	0.203	0.550	0.340	0.420
WATSET[MCL, CW _{top}]	83 092	34 702	473 135	0.097	0.361	0.153	0.351	0.496	0.411
CW _{nolog}	83 092	19 124	672 076	0.087	0.342	0.139	0.364	0.451	0.403
MaxMax	83 092	27 011	461 748	0.176	0.261	0.210	0.582	0.195	0.292
CPM _{k=3}	15 555	4 000	45 231	0.234	0.072	0.111	0.626	0.060	0.110
ECO	83 092	67 645	18 362	0.724	0.034	0.066	0.904	0.002	0.004

Table 4: Results on Russian sorted by F-score on YARN, top three values of each metric are boldfaced.

isolates the neighborhood of polysemous words, which results in the recall drop in comparison to WATSET. CW operates faster due to a simplified update step. On the same graph, CW tends to produce larger clusters than MCL. This leads to a higher recall of “plain” CW as compared to the “plain” MCL, at the cost of lower precision.

Using MCL instead of CW for sense induction in WATSET expectedly produces more fine-grained senses. However, at the global clustering step, these senses erroneously tend to form coarse-grained synsets connecting unrelated senses of the ambiguous words. This explains the generally higher recall of WATSET[MCL, ·]. Despite the randomized nature of CW, variance across runs do not affect the overall ranking: The rank of different versions of CW (*log*, *nolog*, *top*) can change, while the rank of the best CW configuration compared to other methods remains the same.

The MaxMax algorithm shows mixed results. On the one hand, it outputs large clusters uniting more than hundred nodes. This inevitably leads to a high recall, as it is clearly seen in the results for Russian because such synsets still pass

under our cluster size threshold of 150 words. Its synsets on English datasets are even larger and get pruned, which results in low recall. On the other hand, smaller synsets having at most 10–15 words were identified correctly. MaxMax appears to be extremely sensible to edge weighting, which also complicates its practical use.

The CPM algorithm showed unsatisfactory results, emitting giant components encompassing thousands of words. Such clusters were automatically pruned, but the remaining clusters are relatively correctly built synsets, which is confirmed by the high values of precision. When increasing the minimal number of elements in the clique k , recall improves, but at the cost of a dramatic precision drop. We suppose that the network structure assumptions exploited by CPM do not accurately model the structure of our synonymy graphs.

Finally, the ECO method yielded the worst results because the most cluster candidates failed to pass through the constant threshold used for estimating whether a pair of words should be included in the same cluster. Most synsets produced by this method were trivial, i.e., containing only a single

Resource		P	R	F1
BabelNet on WordNet	En	0.729	0.998	0.843
WordNet on BabelNet	En	0.998	0.699	0.822
YARN on RuWordNet	Ru	0.164	0.162	0.163
BabelNet on RuWordNet	Ru	0.348	0.409	0.376
RuWordNet on YARN	Ru	0.670	0.121	0.205
BabelNet on YARN	Ru	0.515	0.109	0.180

Table 5: Performance of lexical resources cross-evaluated against each other.

word. The remaining synsets for both languages have at most three words that have been connected by a chance due to the edge noising procedure used in this method resulting in low recall.

6 Discussion

On the absolute scores. The results obtained on all gold standards (Figure 3) show similar trends in terms of relative ranking of the methods. Yet absolute scores of YARN and RuWordNet are substantially different due to the inherent difference of these datasets. RuWordNet is more domain-specific in terms of vocabulary, so our input set of generic synonymy dictionaries has a limited coverage on this dataset. On the other hand, recall calculated on YARN is substantially higher as this resource was manually built on the basis of synonymy dictionaries used in our experiments.

The reason for low absolute numbers in evaluations is due to an inherent vocabulary mismatch between the input dictionaries of synonyms and the gold datasets. To validate this hypothesis, we performed a cross-resource evaluation presented in Table 5. The low performance of the cross-evaluation of the two resources supports the hypothesis: no single resource for Russian can obtain high recall scores on another one. Surprisingly, even BabelNet, which integrates most of available lexical resources, still does not reach a recall substantially larger than 0.5.²⁰ Note that the results of this cross-dataset evaluation are not directly comparable to results in Table 4 since in our experiments we use much smaller input dictionaries than those used by BabelNet.

On sparseness of the input dictionary. Table 6 presents some examples of the obtained synsets of various sizes for the top WATSET configuration on both languages. As one might observe, the qual-

ity of the results is highly plausible. However, one limitation of all approaches considered in this paper is the dependence on the completeness of the input dictionary of synonyms. In some parts of the input synonymy graph, important bridges between words can be missing, leading to smaller-than-desired synsets. A promising extension of the present methodology is using distributional models to enhance connectivity of the graph by cautiously adding extra relations.

Size	Synset
2	{decimal point, dot}
3	{gullet, throat, food pipe}
4	{microwave meal, ready meal, TV dinner, frozen dinner}
5	{objective case, accusative case, oblique case, object case, accusative}
6	{radio theater, dramatized audiobook, audio theater, radio play, radio drama, audio play}

Table 6: Sample synsets induced by the WATSET[MCL, MCL] method for English.

7 Conclusion

We presented a new robust approach to fuzzy graph clustering that relies on hard graph clustering. Using ego network clustering, the nodes belonging to several local communities are split into several nodes each belonging to one community. The transformed “disambiguated” graph is then clustered using an efficient hard graph clustering algorithm, obtaining a fuzzy clustering as the result. The disambiguated graph facilitates clustering as it contains fewer hubs connecting unrelated nodes from different communities. We apply this meta clustering algorithm to the task of synset induction on two languages, obtaining the best results on three datasets and competitive results on one dataset in terms of F-score as compared to five state-of-the-art graph clustering methods.

Acknowledgments

We acknowledge the support of the Deutsche Forschungsgemeinschaft (DFG) foundation under the “JOIN-T” project, the DAAD, the RFBR under the project no. 16-37-00354 mol_a, and the RFH under the project no. 16-04-12019. We also thank three anonymous reviewers for their helpful comments, Andrew Krizhanovsky for providing a parsed Wiktionary, Natalia Loukachevitch for the provided RuWordNet dataset, and Denis Shirgin who suggested the WATSET name.

²⁰We used BabelNet 3.7 extracting all 3 497 327 synsets that were marked as Russian.

References

- Nikolay Abramov. 1999. *The dictionary of Russian synonyms and semantically related expressions [Slovar' russkikh sinonimov i skhodnykh po smyslu vyrazhenii]*. Russian Dictionaries [Russkie slovari], Moscow, Russia, 7th edition. In Russian.
- Marco Baroni and Alessandro Lenci. 2010. Distributional Memory: A General Framework for Corpus-based Semantics. *Computational Linguistics* 36(4):673–721. https://doi.org/10.1162/coli_a.00016.
- Chris Biemann. 2006. Chinese Whispers: An Efficient Graph Clustering Algorithm and Its Application to Natural Language Processing Problems. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*. Association for Computational Linguistics, New York City, NY, USA, TextGraphs-1, pages 73–80. <http://dl.acm.org/citation.cfm?id=1654774>.
- Chris Biemann. 2012. *Structure Discovery in Natural Language*. Theory and Applications of Natural Language Processing. Springer Berlin Heidelberg. <https://doi.org/10.1007/978-3-642-25923-4>.
- Chris Biemann and Martin Riedl. 2013. Text: now in 2D! A framework for lexical expansion with contextual similarity. *Journal of Language Modelling* 1(1):55–95. <https://doi.org/10.15398/jlm.v1i1.60>.
- Immanuel M. Bomze, Marco Budinich, Panos M. Pardalos, and Marcello Pelillo. 1999. The maximum clique problem. In *Handbook of Combinatorial Optimization*. Springer US, pages 1–74. https://doi.org/10.1007/978-1-4757-3023-4_1.
- Pavel Braslavski, Dmitry Ustalov, Mukhin Mukhin, and Yuri Kiselev. 2016. YARN: Spinning-in-Progress. In *Proceedings of the 8th Global WordNet Conference*. Global WordNet Association, Bucharest, Romania, GWC 2016, pages 58–65. <http://gwc2016.racai.ro/proceedings.pdf>.
- Antonio Di Marco and Roberto Navigli. 2012. Clustering and Diversifying Web Search Results with Graph-Based Word Sense Induction. *Computational Linguistics* 39(3):709–754. https://doi.org/10.1162/COLI_a.00148.
- Vyachelav G. Dikonov. 2013. Development of lexical basis for the Universal Dictionary of UNL Concepts. In *Computational Linguistics and Intellectual Technologies: Papers from the Annual International Conference “Dialogue”*. RGGU, Moscow, volume 12 (19), pages 212–221. <http://www.dialog21.ru/media/1238/dikonov.pdf>.
- Beate Dorow and Dominic Widdows. 2003. Discovering Corpus-Specific Word Senses. In *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics - Volume 2*. Association for Computational Linguistics, Budapest, Hungary, EACL '03, pages 79–82. <https://doi.org/10.3115/1067737.1067753>.
- Martin Everett and Stephen P. Borgatti. 2005. Ego network betweenness. *Social Networks* 27(1):31–38. <https://doi.org/10.1016/j.socnet.2004.11.007>.
- Stefano Faralli, Alexander Panchenko, Chris Biemann, and Simone P. Ponzetto. 2016. Linked Disambiguated Distributional Semantic Networks. In *The Semantic Web – ISWC 2016: 15th International Semantic Web Conference, Kobe, Japan, October 17–21, 2016, Proceedings, Part II*. Springer International Publishing, Cham, pages 56–64. https://doi.org/10.1007/978-3-319-46547-0_7.
- David Gfeller, Jean-Cédric Chappelier, and Paulo De Los Rios. 2005. Synonym Dictionary Improvement through Markov Clustering and Clustering Stability. In *Proceedings of the International Symposium on Applied Stochastic Models and Data Analysis*, pages 106–113. <https://conferences.telecom-bretagne.eu/asmda2005/IMG/pdf/proceedings/106.pdf>.
- Hugo Gonçalo Oliveira and Paolo Gomes. 2014. ECO and Onto.PT: a flexible approach for creating a Portuguese wordnet automatically. *Language Resources and Evaluation* 48(2):373–393. <https://doi.org/10.1007/s10579-013-9249-9>.
- Zhiguo Gong, Chan Wa Cheang, and U. Leong Hou. 2005. Web Query Expansion by WordNet. In *Proceedings of the 16th International Conference on Database and Expert Systems Applications - DEXA '05*, Springer Berlin Heidelberg, Copenhagen, Denmark, pages 166–175. https://doi.org/10.1007/11546924_17.
- Iryna Gurevych, Judith Eckle-Kohler, Silvana Hartmann, Michael Matuschek, Christian M. Meyer, and Christian Wirth. 2012. UBY – A Large-Scale Unified Lexical-Semantic Resource Based on LMF. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Avignon, France, EACL '12, pages 580–590. <http://www.aclweb.org/anthology/E12-1059>.
- Iryna Gurevych, Judith Eckle-Kohler, and Michael Matuschek. 2016. *Linked Lexical Knowledge Bases: Foundations and Applications*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.
- Kris Heylen, Yves Peirsman, Dirk Geeraerts, and Dirk Speelman. 2008. Modelling Word Similarity: an Evaluation of Automatic Synonymy Extraction Algorithms. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation*. European Language Resources Association, Marrakech, Morocco, LREC 2008, pages 3243–3249. http://www.lrec-conf.org/proceedings/lrec2008/pdf/818_paper.pdf.
- David Hope and Bill Keller. 2013. MaxMax: A Graph-Based Soft Clustering Algorithm Applied to Word Sense Induction. In *Computational Linguistics*

- and Intelligent Text Processing: 14th International Conference, CICLing 2013, Samos, Greece, March 24-30, 2013, Proceedings, Part I*, Springer Berlin Heidelberg, Berlin, Heidelberg, pages 368–381. https://doi.org/10.1007/978-3-642-37247-6_30.
- David Jurgens and Ioannis Klapaftis. 2013. SemEval-2013 Task 13: Word Sense Induction for Graded and Non-Graded Senses. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. Association for Computational Linguistics, Atlanta, GA, USA, pages 290–299. <http://www.aclweb.org/anthology/S13-2049>.
- Yuri Kiselev, Sergey V. Porshnev, and Mikhail Mukhin. 2015. Current Status of Russian Electronic Thesauri: Quality, Completeness and Availability [Sovremennoe sostoyanie elektronnykh tezaurusov russkogo jazyka: kachestvo, polnota i dostupnost']. *Programmnaya Ingeneria* 6:34–40. In Russian. http://novtex.ru/prin/full/06_2015.pdf.
- Andrew A. Krizhanovsky and Alexander V. Smirnov. 2013. An approach to automated construction of a general-purpose lexical ontology based on Wiktionary. *Journal of Computer and Systems Sciences International* 52(2):215–225. <https://doi.org/10.1134/S1064230713020068>.
- Cody Kwok, Oren Etzioni, and Daniel S. Weld. 2001. Scaling Question Answering to the Web. *ACM Transactions on Information Systems* 19(3):242–262. <https://doi.org/10.1145/502115.502117>.
- Dekang Lin. 1998. An Information-Theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., Madison, WI, USA, ICML '98, pages 296–304. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.55.1832&rep=rep1&type=pdf>.
- Natalia Loukachevitch. 2011. *Thesauri in information retrieval tasks [Tezaurusy v zadachakh informacionnogo poiska]*. Moscow University Press [Izd-vo MGU], Moscow, Russia. In Russian.
- Natalia V. Loukachevitch, German Lashevich, Anastasia A. Gerasimova, Vladimir V. Ivanov, and Boris V. Dobrov. 2016. Creating Russian WordNet by Conversion. In *Computational Linguistics and Intellectual Technologies: papers from the Annual conference “Dialogue”*. RSUH, Moscow, Russia, pages 405–415. <http://www.dialog-21.ru/media/3409/loukachevitchnetal.pdf>.
- Suresh Manandhar, Ioannis Klapaftis, Dmitriy Dliach, and Sameer Pradhan. 2010. SemEval-2010 Task 14: Word Sense Induction & Disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, Uppsala, Sweden, pages 63–68. <http://www.aclweb.org/anthology/S10-1011>.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, Curran Associates, Inc., Harrahs and Harveys, NV, USA, pages 3111–3119. <https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf>.
- George A. Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM* 38(11):39–41. <https://doi.org/10.1145/219717.219748>.
- Roberto Navigli and Simone P. Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence* 193:217–250. <https://doi.org/10.1016/j.artint.2012.07.001>.
- Gergely Palla, Imre Derenyi, Illes Farkas, and Tamas Vicsek. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435:814–818. <https://doi.org/10.1038/nature03607>.
- Alexander Panchenko. 2011. Comparison of the Baseline Knowledge-, Corpus-, and Web-based Similarity Measures for Semantic Relations Extraction. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*. Association for Computational Linguistics, Edinburgh, UK, pages 11–21. <http://www.aclweb.org/anthology/W11-2502>.
- Alexander Panchenko, Eugen Ruppert, Stefano Faralli, Simone P. Ponzetto, and Chris Biemann. 2017a. Unsupervised Does Not Mean Uninterpretable: The Case for Word Sense Induction and Disambiguation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, Valencia, Spain, pages 86–98. <http://www.aclweb.org/anthology/E17-1009>.
- Alexander Panchenko, Dmitry Ustalov, Nikolay Arefyev, Denis Paperno, Natalia Konstantinova, Natalia Loukachevitch, and Chris Biemann. 2017b. Human and Machine Judgements for Russian Semantic Relatedness. In *Analysis of Images, Social Networks and Texts: 5th International Conference, AIST 2016, Yekaterinburg, Russia, April 7–9, 2016, Revised Selected Papers*. Springer International Publishing, Yekaterinburg, Russia, pages 221–235. https://doi.org/10.1007/978-3-319-52920-2_21.
- Patrick Pantel and Dekang Lin. 2002. Discovering Word Senses from Text. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Edmonton, Alberta, Canada, KDD '02, pages 613–619. <https://doi.org/10.1145/775047.775138>.

Maria Pelevina, Nikolay Arefiev, Chris Biemann, and Alexander Panchenko. 2016. [Making Sense of Word Embeddings](#). In *Proceedings of the 1st Workshop on Representation Learning for NLP*. Association for Computational Linguistics, Berlin, Germany, pages 174–183. <http://anthology.aclweb.org/W16-1620>.

Stijn van Dongen. 2000. *Graph Clustering by Flow Simulation*. Ph.D. thesis, University of Utrecht.

Jean Véronis. 2004. [HyperLex: lexical cartography for information retrieval](#). *Computer Speech & Language* 18(3):223–252. <https://doi.org/10.1016/j.csl.2004.05.002>.

Torsten Zesch, Christof Müller, and Iryna Gurevych. 2008. Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*. European Language Resources Association, Marrakech, Morocco, pages 1646–1652. http://www.lrec-conf.org/proceedings/lrec2008/pdf/420_paper.pdf.

Guangyou Zhou, Yang Liu, Fang Liu, Daojian Zeng, and Jun Zhao. 2013. Improving Question Retrieval in Community Question Answering Using World Knowledge. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*. AAAI Press, Beijing, China, IJCAI '13, pages 2239–2245. <https://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/download/6581/7029>.

A.10 Paper “Unsupervised Semantic Frame Induction using Triclustering”

D. Ustalov, A. Panchenko, A. Kutuzov, C. Biemann, and S. P. Ponzetto, “**Unsupervised Semantic Frame Induction using Triclustering**,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, (Melbourne, Australia), pp. 55–62, Association for Computational Linguistics, July 2018

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/P18-2010>

Unsupervised Semantic Frame Induction using Triclustering

Dmitry Ustalov[†], Alexander Panchenko[‡], Andrei Kutuzov^{*},
Chris Biemann[†], and Simone Paolo Ponzetto[†]

[†]University of Mannheim, Germany

{dmitry, simone}@informatik.uni-mannheim.de

^{*}University of Oslo, Norway

andreku@ifi.uio.no

[‡]University of Hamburg, Germany

{panchenko, biemann}@informatik.uni-hamburg.de

Abstract

We use dependency triples automatically extracted from a Web-scale corpus to perform unsupervised semantic frame induction. We cast the frame induction problem as a *triclustering* problem that is a generalization of clustering for *triadic* data. Our replicable benchmarks demonstrate that the proposed graph-based approach, *Triframes*, shows state-of-the-art results on this task on a FrameNet-derived dataset and performing on par with competitive methods on a verb class clustering task.

1 Introduction

Recent years have seen much work on Frame Semantics (Fillmore, 1982), enabled by the availability of a large set of frame definitions, as well as a manually annotated text corpus provided by the FrameNet project (Baker et al., 1998). FrameNet data enabled the development of wide-coverage frame parsers using supervised learning (Gildea and Jurafsky, 2002; Erk and Padó, 2006; Das et al., 2014, *inter alia*), as well as its application to a wide range of tasks, ranging from answer extraction in Question Answering (Shen and Lapata, 2007) and Textual Entailment (Burchardt et al., 2009; Ben Aharon et al., 2010).

However, frame-semantic resources are arguably expensive and time-consuming to build due to difficulties in defining the frames, their granularity and domain, as well as the complexity of the construction and annotation tasks requiring expertise in the underlying knowledge. Consequently, such resources exist only for a few languages (Boas, 2009) and even English is lacking domain-specific frame-based resources. Possible inroads are cross-lingual semantic annotation transfer (Padó and Lapata, 2009; Hartmann

FrameNet	Role	Lexical Units (LU)
<i>Perpetrator</i>	Subject	kidnapper, alien, militant
<i>FEE</i>	Verb	snatch, kidnap, abduct
<i>Victim</i>	Object	son, people, soldier, child

Table 1: Example of a LU tricluster corresponding to the “Kidnapping” frame from FrameNet.

et al., 2016) or linking FrameNet to other lexical-semantic or ontological resources (Narayanan et al., 2003; Tonelli and Pighin, 2009; Laparra and Rigau, 2010; Gurevych et al., 2012, *inter alia*). But while the arguably simpler task of PropBank-based Semantic Role Labeling has been successfully addressed by unsupervised approaches (Lang and Lapata, 2010; Titov and Klementiev, 2011), fully unsupervised frame-based semantic annotation exhibits far more challenges, starting with the preliminary step of automatically inducing a set of semantic frame definitions that would drive a subsequent text annotation. In this work, we aim at overcoming these issues by automatizing the process of FrameNet construction through unsupervised frame induction techniques.

Triclustering. In this work, we cast the frame induction problem as a *triclustering* task (Zhao and Zaki, 2005; Ignatov et al., 2015), namely a generalization of standard clustering and biclustering (Cheng and Church, 2000), aiming at simultaneously clustering objects along three dimensions (cf. Table 1). First, using triclustering allows to avoid sequential nature of frame induction approaches, e.g. (Kawahara et al., 2014), where two independent clusterings are needed. Second, benchmarking frame induction as triclustering against other methods on dependency triples allows to abstract away the evaluation of the frame induction algorithm from other factors, e.g., the input corpus or pre-processing steps, thus allowing a fair comparison of different induction models.

The **contributions of this paper** are three-fold: (1) we are the *first to apply* triclustering algorithms for unsupervised frame induction, (2) we propose a *new approach to triclustering*, achieving state-of-the-art performance on the frame induction task, (3) we propose a *new method for the evaluation* of frame induction enabling straightforward comparison of approaches. In this paper, we focus on the simplest setup with *subject-verb-object* (SVO) triples and two roles, but our evaluation framework can be extended to more roles.

In contrast to the recent approaches like the one by Jauhar and Hovy (2017), our approach induces semantic frames without any supervision, yet capturing only two core roles: the subject and the object of a frame triggered by verbal predicates. Note that it is not generally correct to expect that the SVO triples obtained by a dependency parser are necessarily the core arguments of a predicate. Such roles can be implicit, i.e., unexpressed in a given context (Schenk and Chiarcos, 2016). Keeping this limitation in mind, we assume that the triples obtained from a Web-scale corpus cover most core arguments sufficiently.

Related Work. *LDA-Frames* (Materna, 2012, 2013) is an approach to inducing semantic frames using LDA (Blei et al., 2003) for generating semantic frames and their respective frame-specific semantic roles at the same time. The authors evaluated their approach against the CPA corpus (Hanks and Pustejovsky, 2005). *ProFinder* (Cheung et al., 2013) is another generative approach that also models both frames and roles as latent topics. The evaluation was performed on the in-domain information extraction task MUC-4 (Sundheim, 1992) and on the text summarization task TAC-2010.¹ Modi et al. (2012) build on top of an unsupervised semantic role labeling model (Titov and Klementiev, 2012). The raw text of sentences from the FrameNet data is used for training. The FrameNet gold annotations are then used to evaluate the labeling of the obtained frames and roles, effectively clustering instances known during induction. Kawahara et al. (2014) harvest a huge collection of verbal predicates along with their argument instances and then apply the Chinese Restaurant Process clustering algorithm to group predicates with similar arguments. The approach was evaluated on the verb

cluster dataset of Korhonen et al. (2003).

A major issue with unsupervised frame induction task is that these and some other related approaches, e.g., (O’Connor, 2013), were all evaluated in completely different incomparable settings, and used different input corpora. In this paper, we propose a methodology to resolve this issue.

2 The Triframes Algorithm

Our approach to frame induction relies on graph clustering. We focused on a simple setup using two roles and the SVO triples, arguing that it still can be useful, as frame roles are primarily expressed by subjects and objects, giving rise to semantic structures extracted in an unsupervised way with high coverage.

Input Data. As the input data, we use SVO triples extracted by a dependency parser. According to our statistics on the dependency-parsed FrameNet corpus of over 150 thousand sentences (Bauer et al., 2012), the SUBJ and OBJ relationships are the two most common shortest paths between frame evoking elements (*FEEs*) and their roles, accounting for 13.5 % of instances of a heavy-tail distribution of over 11 thousand different paths that occur three times or more in the FrameNet data. While this might seem a simplification that does not cover prepositional phrases and frames filling the roles of other frames in a nested fashion, we argue that the overall frame inventory can be induced on the basis of this restricted set of constructions, leaving other paths and more complex instances for further work.

The Method. Our method constructs embeddings for SVO triples to reduce the frame induction problem to a simpler graph clustering problem. Given the vocabulary V , a d -dimensional word embedding model $v \in V \rightarrow \vec{v} \in \mathbb{R}^d$, and a set of SVO triples $T \subseteq V^3$ extracted from a syntactically analyzed corpus, we construct the triple similarity graph \mathcal{G} . Clustering of \mathcal{G} yields sets of triples corresponding to the instances of the semantic frames, thereby clustering frame-evoking predicates and roles simultaneously.

We obtain dense representations of the triples T by concatenating the word vectors corresponding to the elements of each triple by transforming a triple $t = (s, p, o) \in T$ into the $(3d)$ -dimensional vector $\vec{t} = \vec{s} \oplus \vec{p} \oplus \vec{o}$. Subsequently, we use the triple embeddings to generate the undirected graph

¹<https://tac.nist.gov/2010/Summarization>

Algorithm 1 *Triframes* frame induction

Input: an embedding model $v \in V \rightarrow \vec{v} \in \mathbb{R}^d$,
a set of SVO triples $T \subseteq V^3$,
the number of nearest neighbors $k \in \mathbb{N}$,
a graph clustering algorithm CLUSTER.
Output: a set of triframes F .

```
1:  $S \leftarrow \{t \rightarrow \vec{t} \in \mathbb{R}^{3d} : t \in T\}$ 
2:  $E \leftarrow \{(t, t') \in T^2 : t' \in \text{NN}_k^S(\vec{t}), t \neq t'\}$ 
3:  $F \leftarrow \emptyset$ 
4: for all  $C \in \text{CLUSTER}(T, E)$  do
5:    $f_s \leftarrow \{s \in V : (s, v, o) \in C\}$ 
6:    $f_v \leftarrow \{v \in V : (s, v, o) \in C\}$ 
7:    $f_o \leftarrow \{o \in V : (s, v, o) \in C\}$ 
8:    $F \leftarrow F \cup \{(f_s, f_v, f_o)\}$ 
9: return  $F$ 
```

$\mathcal{G} = (T, E)$ by constructing the edge set $E \subseteq T^2$. For that, we compute $k \in \mathbb{N}$ nearest neighbors of each triple vector $\vec{t} \in \mathbb{R}^{3d}$ and establish cosine similarity-weighted edges between the corresponding triples.

Then, we assume that the triples representing similar contexts appear in similar roles, which is explicitly encoded by the concatenation of the corresponding vectors of the words constituting the triple. We use graph clustering of \mathcal{G} to retrieve communities of similar triples forming frame clusters; a clustering algorithm is a function $\text{CLUSTER} : (T, E) \rightarrow \mathbb{C}$ such that $T = \bigcup_{C \in \mathbb{C}} C$. Finally, for each cluster $C \in \mathbb{C}$, we aggregate the subjects, the verbs, and the objects of the contained triples into separate sets. As the result, each cluster is transformed into a *triframe*, which is a triple that is composed of the subjects $f_s \subseteq V$, the verbs $f_v \subseteq V$, and the objects $f_o \subseteq V$.

Our frame induction approach outputs a set of triframes F as presented in Algorithm 1. The hyper-parameters of the algorithm are the number of nearest neighbors for establishing edges (k) and the graph clustering algorithm CLUSTER. During the concatenation of the vectors for words forming triples, the $(|T| \times 3d)$ -dimensional vector space S is created. Thus, given the triple $t \in T$, we denote the k nearest neighbors extraction procedure of its concatenated embedding from S as $\text{NN}_k^S(\vec{t}) \subseteq T$. We used $k = 10$ nearest neighbors per triple.

To cluster the nearest neighbor graph of SVO triples \mathcal{G} , we use the WATSET *fuzzy graph clustering* algorithm (Ustalov et al., 2017). It treats the vertices T of the input graph \mathcal{G} as the SVO triples, induces their senses, and constructs an intermedi-

ate sense-aware representation that is clustered using the Chinese Whispers (CW) hard clustering algorithm (Biemann, 2006). We chose WATSET due to its performance on the related synset induction task, its fuzzy nature, and the ability to find the number of frames automatically.

3 Evaluation

Input Corpus. In our evaluation, we use triple frequencies from the DepCC dataset (Panchenko et al., 2018), which is a dependency-parsed version of the Common Crawl corpus, and the standard 300-dimensional word embeddings model trained on the Google News corpus (Mikolov et al., 2013). All evaluated algorithms are executed on the same set of triples, eliminating variations due to different corpora or pre-processing.

Datasets. We cast the complex multi-stage frame induction task as a straightforward triple clustering task. We constructed a gold standard set of triclusters, each corresponding to a FrameNet frame, similarly to the one illustrated in Table 1. To construct the evaluation dataset, we extracted frame annotations from the over 150 thousand sentences from the FrameNet 1.7 (Baker et al., 1998). Each sentence contains data about the frame, FEE, and its arguments, which were used to generate triples in the form $(\text{word}_i : \text{role}_1, \text{word}_j : \text{FEE}, \text{word}_k : \text{role}_2)$, where $\text{word}_{i/j/k}$ correspond to the roles and FEE in the sentence. We omitted roles expressed by multiple words as we use dependency parses, where one node represents a single word only.

For the sentences where more than two roles are present, all possible triples were generated. Sentences with less than two roles were omitted. Finally, for each frame, we selected only two roles, which are most frequently co-occurring in the FrameNet annotated texts. This has left us with about 100 thousand instances for the evaluation. For the evaluation purposes, we operate on the intersection of triples from DepCC and FrameNet. Experimenting on the full set of DepCC triples is only possible for several methods that scale well (WATSET, CW, k -means), but is prohibitively expensive for other methods (LDA-Frames, NOAC).

In addition to the frame induction evaluation, where subjects, objects, and verbs are evaluated together, we also used a dataset of polysemous verb classes introduced in (Korhonen et al., 2003) and employed by Kawahara et al. (2014). Statis-

Dataset	# instances	# unique	# clusters
FrameNet Triples	99,744	94,170	383
Poly. Verb Classes	246	110	62

Table 2: Statistics of the evaluation datasets.

tics of both datasets are summarized in Table 2. Note that the polysemous verb dataset is rather small, whereas the FrameNet triples set is fairly large, enabling reliable comparisons.

Evaluation Measures. Following the approach for verb class evaluation by Kawahara et al. (2014), we employ *normalized modified purity* (nmPU) and *normalized inverse purity* (niPU) as the clustering quality measures. Given the set of the obtained clusters K and the set of the gold clusters G , normalized modified purity quantifies the clustering precision as the average of the weighted overlap $\delta_{K_i}(K_i \cap G_j)$ between each cluster $K_i \in K$ and the gold cluster $G_j \in G$ that maximizes the overlap with K_i : $\text{nmPU} = \frac{1}{N} \sum_{i=1}^{|K|} \max_{\substack{j=1 \\ |K_i| > 1}} \max_{1 \leq j \leq |G|} \delta_{K_i}(K_i \cap G_j)$, where the weighted overlap is the sum of the weights c_{iv} for each word v in i -th cluster: $\delta_{K_i}(K_i \cap G_j) = \sum_{v \in K_i \cap G_j} c_{iv}$. Note that nmPU counts all the singleton clusters as wrong. Similarly, normalized inverse purity (collocation) quantifies the clustering recall: $\text{niPU} = \frac{1}{N} \sum_{j=1}^{|G|} \max_{1 \leq i \leq |K|} \delta_{G_j}(K_i \cap G_j)$. nmPU and niPU are combined together as the harmonic mean to yield the overall clustering F-score (F_1), which we use to rank the approaches.

Our framework can be extended to evaluation of more than two roles by generating more roles per frame. Currently, given a set of gold triples generated from the FrameNet, each triple element has a role, e.g., “*Victim*”, “*Predator*”, and “*FEE*”. We use fuzzy clustering evaluation measure which operates not on triples, but instead on a set of tuples. Consider for instance a gold triple (Freddy: *Predator*, kidnap: *FEE*, kid: *Victim*). It will be converted to three pairs (Freddy, *Predator*), (kidnap, *FEE*), (kid, *Victim*). Each cluster in both K and G is transformed into a union of all constituent typed pairs. The quality measures are finally calculated between these two sets of tuples, K , and G . Note that one can easily pull in more than two core roles by adding to this gold standard set of tuples other roles of the frame, e.g., (forest, *Location*). In our experiments, we focused on two main roles as our contribution is related to the application of triclusetering methods. However, if more advanced

methods of clustering are used, yielding clusters of arbitrary modality (n -clustering), one could also use our evaluation schema.

Baselines. We compare our method to several available state-of-the-art baselines applicable to our dataset of triples.

LDA-Frames by Materna (2012, 2013) is a frame induction method based on topic modeling. We ran 500 iterations of the model with the default parameters. *Higher-Order Skip-Gram (HOSG)* by Cotterell et al. (2017) generalizes the Skip-Gram model (Mikolov et al., 2013) by extending it from word-context co-occurrence matrices to tensors factorized with a polyadic decomposition. In our case, this tensor consisted of SVO triple counts. We trained three vector arrays (for subjects, verbs and objects) on the 108,073 SVO triples from the *FrameNet* corpus, using the implementation by the authors. Training was performed with 5 negative samples, 300-dimensional vectors, and 10 epochs. We constructed an embedding of a triple by concatenating embeddings for subjects, verbs, and objects, and clustered them using k -means with the number of clusters set to 10,000 (this value provided the best performance). *NOAC* (Egurnov et al., 2017) is an extension of the Object Attribute Condition (OAC) triclusetering algorithm (Ignatov et al., 2015) to numerically weighted triples. This incremental algorithm searches for dense regions in triadic data. A minimum density of 0.25 led to the best results. In the *Triadic* baselines, independent word embeddings of subject, object, and verb are concatenated and then clustered using a *hard clustering algorithm*: k -means, spectral clustering, or CW.

We tested various hyper-parameters of each of these algorithms and report the best results overall per clustering algorithm. Two trivial baselines are *Singlets* that creates a single cluster per instance and *Whole* that creates one cluster for all elements.

4 Results

We perform two experiments to evaluate our approach: (1) a frame induction experiment on the FrameNet annotated corpus by Bauer et al. (2012); (2) the polysemous verb clustering experiment on the dataset by Korhonen et al. (2003). The first is based on the newly introduced frame induction evaluation schema (cf. Section 3). The second one evaluates the quality of verb clusters only on a standard dataset from prior work.

Method	Verb			Subject			Object			Frame		
	nmPU	niPU	F ₁									
Triframes WATSET	42.84	88.35	57.70	54.22	81.40	65.09	53.04	83.25	64.80	55.19	60.81	57.87
HOSG (Cotterell et al., 2017)	44.41	68.43	53.86	52.84	74.53	61.83	54.73	74.05	62.94	55.74	50.45	52.96
NOAC (Egurnov et al., 2017)	20.73	88.38	33.58	57.00	80.11	66.61	57.32	81.13	67.18	44.01	63.21	51.89
Triadic Spectral	49.62	24.90	33.15	50.07	41.07	45.13	50.50	41.82	45.75	52.05	28.60	36.91
Triadic <i>k</i> -Means	63.87	23.16	33.99	63.15	38.20	47.60	63.98	37.43	47.23	63.64	24.11	34.97
LDA-Frames (Materna, 2013)	26.11	66.92	37.56	17.28	83.26	28.62	20.80	90.33	33.81	18.80	71.17	29.75
Triframes CW	7.75	6.48	7.06	3.70	14.07	5.86	51.91	76.92	61.99	21.67	26.50	23.84
Singletons	0.00	25.23	0.00	0.00	25.68	0.00	0.00	20.80	0.00	32.34	22.15	26.29
Whole	3.62	100.0	6.98	2.41	98.41	4.70	2.38	100.0	4.64	2.63	99.55	5.12

Table 3: Frame evaluation results on the triples from the FrameNet 1.7 corpus (Baker et al., 1998). The results are sorted by the descending order of the Frame F₁-score. Best results are boldfaced.

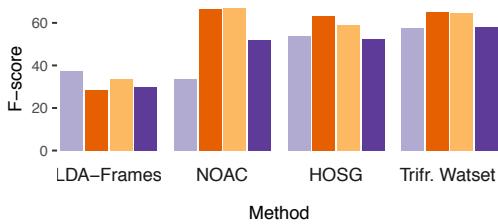


Figure 1: F₁-scores for verbs, subjects, objects, frames corresponding to Table 3.

Frame Induction Experiment. In Table 3 and Figure 1, the results of the experiment are presented. Triframes based on WATSET clustering outperformed the other methods on both Verb F₁ and overall Frame F₁. The HOSG-based clustering proved to be the most competitive baseline, yielding decent scores according to all four measures. The NOAC approach captured the frame grouping of slot fillers well but failed to establish good verb clusters. Note that NOAC and HOSG use only the graph of syntactic triples and do not rely on pre-trained word embeddings. This suggests a high complementarity of signals based on distributional similarity and global structure of the triple graph. Finally, the simpler *Triadic* baselines relying on hard clustering algorithms showed low performance, similar to that of *LDA-Frames*, justifying the more elaborate WATSET method.

While triples are intuitively less ambiguous than words, still some frequent and generic triples like (she, make, it) can act as hubs in the graph, making it difficult to split it into semantically plausible clusters. The poor results of the Chinese Whispers hard clustering algorithm illustrate this. Since the hubs are ambiguous, i.e., can belong to multiple clusters, the use of the WATSET fuzzy clustering algorithm that splits the hubs by disambiguating them leads to the best results (see Table 3).

Method	nmPU	niPU	F ₁
LDA-Frames	52.60	45.84	48.98
Triframes WATSET	40.05	62.09	48.69
NOAC	37.19	64.09	47.07
HOSG	38.22	43.76	40.80
Triadic Spectral	35.76	38.96	36.86
Triadic <i>k</i> -Means	52.22	27.43	35.96
Triframes CW	18.05	12.72	14.92
Whole	24.14	79.09	36.99
Singletons	0.00	27.21	0.00

Table 4: Evaluation results on the dataset of polysemous verb classes by Korhonen et al. (2003).

Verb Clustering Experiment. Table 4 presents results on the second dataset for the best models identified on the first dataset. The *LDA-Frames* yielded the best results with our approach performing comparably in terms of the F₁-score. We attribute the low performance of the Triframes method based on CW clustering to its hard partitioning output, whereas the evaluation dataset contains fuzzy clusters. Different rankings also suggest that frame induction cannot simply be treated as a verb clustering and requires a separate task.

5 Conclusion

In this paper, we presented the first application of *triclusetering* for unsupervised *frame induction*. We designed a dataset based on the FrameNet and SVO triples to enable fair corpus-independent evaluations of frame induction algorithms. We tested several triclusetering methods as the baselines and proposed a new graph-based triclusetering algorithm that yields state-of-the-art results. A promising direction for future work is using the induced frames in applications, such as Information Extraction and Question Answering.

Additional illustrations and examples of extracted frames are available in the supplementary materials. The source code and the data are available online under a permissive license.²

²<https://github.com/uhh-lt/triframes>

Acknowledgments

We acknowledge the support of DFG under the “JOIN-T” and “ACQuA” projects and thank three anonymous reviewers for their helpful comments. Furthermore, we thank Dmitry Egurnov, Dmitry Ignatov, and Dmitry Gnatyshak for help in operating the NOAC method using the multimodal clustering toolbox. Besides, we are grateful to Ryan Cotterell and Adam Poliak for a discussion and an implementation of the HOSG method. Finally, we thank Bonaventura Coppolla for discussions and preliminary work on graph-based frame induction.

References

- Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. [The Berkeley FrameNet Project](#). In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL ’98, pages 86–90, Montreal, QC, Canada. Association for Computational Linguistics.
- Daniel Bauer, Hagen Fürstenau, and Owen Ramshaw. 2012. [The Dependency-Parsed FrameNet Corpus](#). In *Proceedings of the Eight International Conference on Language Resources and Evaluation*, LREC 2012, pages 3861–3867, Istanbul, Turkey. European Language Resources Association (ELRA).
- Roni Ben Aharon, Idan Szpektor, and Ido Dagan. 2010. [Generating Entailment Rules from FrameNet](#). In *Proceedings of the ACL 2010 Conference Short Papers*, pages 241–246, Uppsala, Sweden. Association for Computational Linguistics.
- Chris Biemann. 2006. [Chinese Whispers: An Efficient Graph Clustering Algorithm and Its Application to Natural Language Processing Problems](#). In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*, TextGraphs-1, pages 73–80, New York, NY, USA. Association for Computational Linguistics.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. [Latent Dirichlet Allocation](#). *Journal of Machine Learning Research*, 3:993–1022.
- Hans C. Boas. 2009. *Multilingual FrameNets in Computational Lexicography: Methods and Applications*. Trends in Linguistics. Studies and Monographs. Mouton de Gruyter.
- Aljoscha Burchardt, Marco Pennacchiotti, Stefan Thater, and Manfred Pinkal. 2009. [Assessing the impact of frame semantics on textual entailment](#). *Natural Language Engineering*, 15(4):527–550.
- Yizong Cheng and George M. Church. 2000. [Bioclustering of Expression Data](#). In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 93–103. AAAI Press.
- Jackie C. K. Cheung, Hoifung Poon, and Lucy Vanderwende. 2013. [Probabilistic Frame Induction](#). In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 837–846, Atlanta, GA, USA. Association for Computational Linguistics.
- Ryan Cotterell, Adam Poliak, Benjamin Van Durme, and Jason Eisner. 2017. [Explaining and Generalizing Skip-Gram through Exponential Family Principal Component Analysis](#). In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 175–181, Valencia, Spain. Association for Computational Linguistics.
- Dipanjan Das, Desai Chen, André F. T. Martins, Nathan Schneider, and Noah A. Smith. 2014. [Frame-Semantic Parsing](#). *Computational Linguistics*, 40(1):9–56.
- Dmitry Egurnov, Dmitry Ignatov, and Engelbert M. Nguifo. 2017. [Mining Triclusters of Similar Values in Triadic Real-Valued Contexts](#). In *14th International Conference on Formal Concept Analysis - Supplementary Proceedings*, pages 31–47, Rennes, France.
- Katrin Erk and Sebastian Padó. 2006. [SHALMANESER — A Toolchain For Shallow Semantic Parsing](#). In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, LREC 2006, pages 527–532, Genoa, Italy. European Language Resources Association (ELRA).
- Charles J. Fillmore. 1982. Frame Semantics. In *Linguistics in the Morning Calm*, pages 111–137. Hanshin Publishing Co., Seoul, South Korea.
- Daniel Gildea and Martin Jurafsky. 2002. [Automatic Labeling of Semantic Roles](#). *Computational Linguistics*, 28(3):245–288.
- Iryna Gurevych, Judith Eckle-Kohler, Silvana Hartmann, Michael Matuschek, Christian M. Meyer, and Christian Wirth. 2012. [UBY – A Large-Scale Unified Lexical-Semantic Resource Based on LMF](#). In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL ’12, pages 580–590, Avignon, France. Association for Computational Linguistics.
- Patrick Hanks and James Pustejovsky. 2005. [A Pattern Dictionary for Natural Language Processing](#). *Revue Française de linguistique appliquée*, 10(2):63–82.
- Silvana Hartmann, Judith Eckle-Kohler, and Iryna Gurevych. 2016. [Generating Training Data for Semantic Role Labeling based on Label Transfer from Linked Lexical Resources](#). *Transactions of the Association for Computational Linguistics*, 4:197–213.

- Dmitry I. Ignatov, Dmitry V. Gnatyshak, Sergei O. Kuznetsov, and Boris G. Mirkin. 2015. *Triadic Formal Concept Analysis and triclusustering: searching for optimal patterns*. *Machine Learning*, 101(1-3):271–302.
- Sujay Kumar Jauhar and Eduard Hovy. 2017. *Embedded Semantic Lexicon Induction with Joint Global and Local Optimization*. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 209–219, Vancouver, Canada. Association for Computational Linguistics.
- Daisuke Kawahara, Daniel W. Peterson, and Martha Palmer. 2014. *A Step-wise Usage-based Method for Inducing Polysemy-aware Verb Classes*. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics Volume 1: Long Papers*, ACL 2014, pages 1030–1040, Baltimore, MD, USA. Association for Computational Linguistics.
- Anna Korhonen, Yuval Krymolowski, and Zvika Marx. 2003. *Clustering Polysemic Subcategorization Frame Distributions Semantically*. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL ’03, pages 64–71, Sapporo, Japan. Association for Computational Linguistics.
- Joel Lang and Mirella Lapata. 2010. *Unsupervised Induction of Semantic Roles*. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 939–947, Los Angeles, CA, USA. Association for Computational Linguistics.
- Egoitz Laparra and German Rigau. 2010. *eXtended WordFrameNet*. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, LREC 2010, pages 1214–1219, Valletta, Malta. European Language Resources Association (ELRA).
- Jiří Materna. 2012. *LDA-Frames: An Unsupervised Approach to Generating Semantic Frames*. In *Computational Linguistics and Intelligent Text Processing, Proceedings, Part I*, CICLing 2012, pages 376–387, New Delhi, India. Springer Berlin Heidelberg.
- Jiří Materna. 2013. *Parameter Estimation for LDA-Frames*. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 482–486, Atlanta, GA, USA. Association for Computational Linguistics.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. *Distributed Representations of Words and Phrases and their Compositionality*. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., Harrahs and Harveys, NV, USA.
- Ashutosh Modi, Ivan Titov, and Alexandre Klementiev. 2012. *Unsupervised Induction of Frame-Semantic Representations*. In *Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure*, pages 1–7, Montréal, Canada. Association for Computational Linguistics.
- Srini Narayanan, Collin Baker, Charles Fillmore, and Miriam Petrucci. 2003. *FrameNet Meets the Semantic Web: Lexical Semantics for the Web*. In *The Semantic Web - ISWC 2003: Second International Semantic Web Conference, Sanibel Island, FL, USA, October 20-23, 2003. Proceedings*, pages 771–787, Heidelberg, Germany. Springer Berlin Heidelberg.
- Brendan O’Connor. 2013. *Learning Frames from Text with an Unsupervised Latent Variable Model*. arXiv preprint arXiv:1307.7382.
- Sebastian Padó and Mirella Lapata. 2009. *Cross-lingual Annotation Projection of Semantic Roles*. *Journal of Artificial Intelligence Research*, 36(1):307–340.
- Alexander Panchenko, Eugen Ruppert, Stefano Faralli, Simone Paolo Ponzetto, and Chris Biemann. 2018. *Building a Web-Scale Dependency-Parsed Corpus from Common Crawl*. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, LREC 2018, pages 1816–1823, Miyazaki, Japan. European Language Resources Association (ELRA).
- Niko Schenk and Christian Chiarcos. 2016. *Unsupervised Learning of Prototypical Fillers for Implicit Semantic Role Labeling*. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1473–1479, San Diego, CA, USA. Association for Computational Linguistics.
- Dan Shen and Mirella Lapata. 2007. *Using Semantic Roles to Improve Question Answering*. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 12–21, Prague, Czech Republic. Association for Computational Linguistics.
- Beth M. Sundheim. 1992. *Overview of the Fourth Message Understanding Evaluation and Conference*. In *Proceedings of the 4th Conference on Message Understanding*, MUC4 ’92, pages 3–21, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ivan Titov and Alexandre Klementiev. 2011. *A Bayesian Model for Unsupervised Semantic Parsing*. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1445–1455, Portland, OR, USA. Association for Computational Linguistics.

Ivan Titov and Alexandre Klementiev. 2012. **A Bayesian Approach to Unsupervised Semantic Role Induction**. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 12–22, Avignon, France. Association for Computational Linguistics.

Sara Tonelli and Daniele Pighin. 2009. **New Features for FrameNet - WordNet Mapping**. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 219–227, Boulder, CO, USA. Association for Computational Linguistics.

Dmitry Ustalov, Alexander Panchenko, and Chris Biemann. 2017. **Watset: Automatic Induction of Synsets from a Graph of Synonyms**. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2017, pages 1579–1590, Vancouver, Canada. Association for Computational Linguistics.

Lizhuang Zhao and Mohammed J. Zaki. 2005. **TRI-CLUSTER: An Effective Algorithm for Mining Coherent Clusters in 3D Microarray Data**. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, SIGMOD '05*, pages 694–705, New York, NY, USA. ACM.

A.11 Paper “Making Sense of Word Embeddings”

M. Pelevina, N. Arefiev, C. Biemann, and A. Panchenko, “**Making Sense of Word Embeddings**,” in *Proceedings of the 1st Workshop on Representation Learning for NLP*, (Berlin, Germany), pp. 174–183, Association for Computational Linguistics, Aug. 2016

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/W16-1620>

Making Sense of Word Embeddings

Maria Pelevina¹, Nikolay Arefyev², Chris Biemann¹ and Alexander Panchenko¹

¹Technische Universität Darmstadt, LT Group, Computer Science Department, Germany

²Moscow State University, Faculty of Computational Mathematics and Cybernetics, Russia

panchenko@lt.informatik.tu-darmstadt.de

Abstract

We present a simple yet effective approach for learning word sense embeddings. In contrast to existing techniques, which either directly learn sense representations from corpora or rely on sense inventories from lexical resources, our approach can induce a sense inventory from existing word embeddings via clustering of ego-networks of related words. An integrated WSD mechanism enables labeling of words in context with learned sense vectors, which gives rise to downstream applications. Experiments show that the performance of our method is comparable to state-of-the-art unsupervised WSD systems.

1 Introduction

Term representations in the form of dense vectors are useful for many natural language processing applications. First of all, they enable the computation of semantically related words. Besides, they can be used to represent other linguistic units, such as phrases and short texts, reducing the inherent sparsity of traditional vector-space representations (Salton et al., 1975).

One limitation of most word vector models, including sparse (Baroni and Lenci, 2010) and dense (Mikolov et al., 2013) representations, is that they conflate all senses of a word into a single vector. Several architectures for learning multi-prototype embeddings were proposed that try to address this shortcoming (Huang et al., 2012; Tian et al., 2014; Neelakantan et al., 2014; Nieto Piña and Johansson, 2015; Bartunov et al., 2016). Li and Jurafsky (2015) provide indications that such sense vectors improve the performance of text pro-

cessing applications, such as part-of-speech tagging and semantic relation identification.

The contribution of this paper is a novel method for learning word sense vectors. In contrast to previously proposed methods, our approach relies on existing single-prototype word embeddings, transforming them to sense vectors via ego-network clustering. An ego network consists of a single node (ego) together with the nodes they are connected to (alters) and all the edges among those alters. Our method is fitted with a word sense disambiguation (WSD) mechanism, and thus words in context can be mapped to these sense representations. An advantage of our method is that one can use existing word embeddings and/or existing word sense inventories to build sense embeddings. Experiments show that our approach performs comparably to state-of-the-art unsupervised WSD systems.

2 Related Work

Our method learns multi-prototype word embeddings and applies them to WSD. Below we briefly review both strains of research.

2.1 Multi-Prototype Word Vector Spaces

In his pioneering work, Schütze (1998) induced sparse sense vectors by clustering context vectors using the EM algorithm. This approach is fitted with a similarity-based WSD mechanism. Later, Reisinger and Mooney (2010) presented a multi-prototype vector space. Sparse TF-IDF vectors are clustered using a parametric method fixing the same number of senses for all words. Sense vectors are centroids of the clusters.

While most dense word vector models represent a word with a single vector and thus conflate senses (Mikolov et al., 2013; Pennington et al., 2014), there are several approaches that produce word sense embeddings. Huang et al. (2012) learn

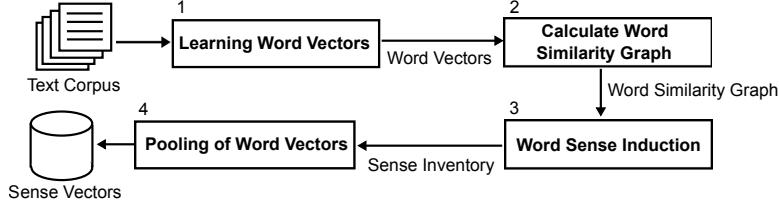


Figure 1: Schema of the word sense embeddings learning method.

dense vector spaces with neural networks. First, contexts are represented with word embeddings and clustered. Second, word occurrences are re-labeled in the corpus according to the cluster they belong to. Finally, embeddings are re-trained on this sense-labeled terms. Tian et al. (2014) introduced a probabilistic extension of the Skip-gram model (Mikolov et al., 2013) that learns multiple sense-aware prototypes weighted by their prior probability. These models use parametric clustering algorithms that produce a fixed number of senses per word.

Neelakantan et al. (2014) proposed a multi-sense extension of the Skip-gram model that was the first one to learn the number of senses by itself. During training, a new sense vector is allocated if the current context’s similarity to existing senses is below some threshold. Li and Jurafsky (2015) use a similar idea by integrating the Chinese Restaurant Process into the Skip-gram model. All mentioned above sense embeddings were evaluated on the contextual word similarity task, each one improving upon previous models.

Nieto and Johansson (2015) presented another multi-prototype modification of the Skip-gram model. Their approach outperforms that of Neelakantan et al. (2014), but requires as an input the number of senses for each word.

Li and Jurafsky (2015) show that sense embeddings can significantly improve the performance of part-of-speech tagging, semantic relation identification and semantic relatedness tasks, but yield no improvement for named entity recognition and sentiment analysis.

Bartunov et al. (2016) introduced AdaGram, a non-parametric method for learning sense embeddings based on a Bayesian extension of the Skip-gram model. The granularity of learned sense embeddings is controlled by the parameter α . Comparisons of their approach to (Neelakantan et al., 2014) on three SemEval word sense induction and

disambiguation datasets show the advantage of their method. For this reason, we use AdaGram as a representative of the state-of-the-art methods in our experiments.

Several approaches rely on a knowledge base (KB) to provide sense information. Bordes et al. (2011) propose a general method to represent entities of any KB as a dense vector. Such representation helps to integrate KBs into NLP systems. Another approach that uses sense inventories of knowledge bases was presented by Camacho-Collados et al. (2015). Rothe and Schütze (2015) combined word embeddings on the basis of WordNet synsets to obtain sense embeddings. The approach is evaluated on lexical sample tasks by adding synset embeddings as features to an existing WSD system. They used a weighted pooling similar to the one we use, but their method is not able to find new senses in a corpus. Finally, Nieto Piña and Johansson (2016) used random walks on the Swedish Wordnet to generate training data for the Skip-gram model.

2.2 Word Sense Disambiguation (WSD)

Many different designs of WSD systems were proposed, see (Agirre and Edmonds, 2007; Navigli, 2009). Supervised approaches use an explicitly sense-labeled training corpus to construct a model, usually building one model per target word (Lee and Ng, 2002; Klein et al., 2002). These approaches demonstrate top performance in competitions, but require considerable amounts of sense-labeled examples.

Knowledge-based approaches do not learn a model per target, but rather derive sense representation from information available in a lexical resource, such as WordNet. Examples of such system include (Lesk, 1986; Banerjee and Pedersen, 2002; Pedersen et al., 2005; Moro et al., 2014).

Unsupervised WSD approaches rely neither on hand-annotated sense-labeled corpora, nor on

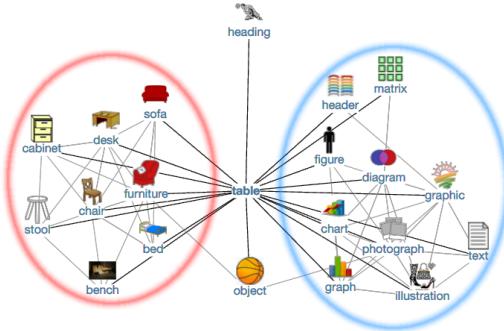


Figure 2: Visualization of the ego-network of “table” with furniture and data sense clusters. Note that the target “table” is excluded from clustering.

handcrafted lexical resources. Instead, they automatically induce a sense inventory from raw corpora. Such unsupervised sense induction methods fall into two categories: context clustering, such as (Pedersen and Bruce, 1997; Schütze, 1998; Reisinger and Mooney, 2010; Neelakantan et al., 2014; Bartunov et al., 2016) and word (ego-network) clustering, such as (Lin, 1998; Pantel and Lin, 2002; Widdows and Dorow, 2002; Biemann, 2006; Hope and Keller, 2013). Unsupervised methods use disambiguation clues from the induced sense inventory for word disambiguation. Usually, the WSD procedure is determined by the design of sense inventory. It might be the highest overlap between the instance’s context words and the words of the sense cluster, as in (Hope and Keller, 2013) or the smallest distance between context words and sense hubs in graph sense representation, as in (Véronis, 2004).

3 Learning Word Sense Embeddings

Our method consists of the four main stages depicted in Figure 1: (1) learning word embeddings; (2) building a graph of nearest neighbours based on vector similarities; (3) induction of word senses using ego-network clustering; and (4) aggregation of word vectors with respect to the induced senses.

Our method can use existing word embeddings, sense inventories and word similarity graphs. To demonstrate such use-cases and to study the performance of the method in different settings, as variants of the complete pipeline presented in Figure 1, we experiment with two additional setups. First, we use an alternative approach to compute

the word similarity graph, which relies on dependency features and is expected to provide more accurate similarities (therefore, the stage (2) is changed). Second, we use a sense inventory constructed using crowdsourcing (thus, stages (2) and (3) are skipped). Below we describe each of the stages of our method in detail.

3.1 Learning Word Vectors

To learn word vectors, we use the *word2vec* toolkit (Mikolov et al., 2013), namely we train CBOW word embeddings with 100 or 300 dimensions, context window size of 3 and minimum word frequency of 5. We selected these parameters according to prior evaluations, e.g. (Baroni et al., 2014), and tested them on the development dataset (see Section 5.1). Initial experiments showed that this configuration is superior to others, e.g. the Skip-gram model, with respect to WSD performance.

For training, we modified the standard implementation of *word2vec*¹ so that it also saves context vectors needed for one of our WSD approaches. For experiments, we use two commonly used corpora for training distributional models: Wikipedia² and ukWaC (Ferraresi et al., 2008).

3.2 Calculating Word Similarity Graph

At this step, we build a graph of word similarities, such as (table, desk, 0.78). For each word we retrieve its 200 nearest neighbours. This number is motivated by prior studies (Biemann and Riedl, 2013; Panchenko, 2013): as observed, only few words have more strongly semantically related words. This graph is computed either based on word embeddings learned during the previous step or using semantic similarities provided by the *JoBimText* framework (Biemann and Riedl, 2013).

Similarities using word2vec (w2v). In this case, nearest neighbours of a term are terms with the highest cosine similarity of their respective vectors. For scalability reasons, we perform similarity computations via block matrix multiplications, using blocks of 1000 vectors.

Similarities using JoBimText (JBT). In this unsupervised approach, every word is represented

¹<https://code.google.com/p/word2vec>

²We used an English Wikipedia dump of October 2015: <http://panchenko.me/data/joint/corpora/en59g/wikipedia.txt.gz>

as a bag of sparse dependency-based features extracted using the Malt parser and collapsed using an approach similar to (Ruppert et al., 2015). Features are normalized using the LMI score (Church and Hanks, 1990) and further pruned down according to the recommended defaults: we keep 1000 features per word and 1000 words per feature. Similarity of two words is equal to the number of common features.

Multiple alternatives exist for computation of semantic relatedness (Zhang et al., 2013). JBT has two advantages in our case: (1) accurate estimation of word similarities based on dependency features; (2) efficient computation of nearest neighbours for all words in a corpus. Besides, we observed that nearest neighbours of word embeddings often tend to belong to the dominant sense, even if minor senses have significant support in the training corpus. We wanted to test if the same problem remains for a principally different method for similarity computation.

Algorithm 1: Word sense induction.

input : T – word similarity graph, N – ego-network size, n – ego-network connectivity, k – minimum cluster size
output: for each term $t \in T$, a clustering S_t of its N most similar terms

foreach $t \in T$ **do**

- $V \leftarrow N$ most similar terms of t from T
- $G \leftarrow$ graph with V as nodes and no edges E
- foreach** $v \in V$ **do**

 - $V' \leftarrow n$ most similar terms of v from T
 - foreach** $v' \in V'$ **do**

 - | if $v' \in V$ **then** add edge (v, v') to E

 - end**

- end**
- $S_t \leftarrow$ ChineseWhispers(G)
- $S_t \leftarrow \{s \in S_t : |s| \geq k\}$

end

3.3 Word Sense Induction

We induce a sense inventory using a method similarly to (Pantel and Lin, 2002) and (Biemann, 2006). A word sense is represented by a word cluster. For instance the cluster “chair, bed, bench, stool, sofa, desk, cabinet” can represent the sense “table (furniture)”. To induce senses, first we construct an ego-network G of a word t and then perform graph clustering of this network. The iden-

Vector	Nearest Neighbours
table	tray, bottom, diagram, bucket, brackets, stack, basket, list, parenthesis, cup, trays, pile, play-field, bracket, pot, drop-down, cue, plate
table#0	leftmost#0, column#1, randomly#0, tableau#1, top-left0, indent#1, bracket#3, pointer#0, footer#1, cursor#1, diagram#0, grid#0
table#1	pile#1, stool#1, tray#0, basket#0, bowl#1, bucket#0, box#0, cage#0, saucer#3, mirror#1, birdcage#0, hole#0, pan#1, lid#0

Table 1: Neighbours of the word “table” and its senses produced by our method. The neighbours of the initial vector belong to both senses, while those of sense vectors are sense-specific.

tified clusters are interpreted as senses (see Table 2). Words referring to the same sense tend to be tightly connected, while having fewer connections to words referring to different senses.

The sense induction presented in Algorithm 1 processes one word t of the word similarity graph T per iteration. First, we retrieve nodes V of the ego-network G : these are the N most similar words of t according to T . The target word t itself is not part of the ego-network. Second, we connect the nodes in G to their n most similar words from T . Finally, the ego-network is clustered with the Chinese Whispers algorithm (Biemann, 2006). This method is parameter free, thus we make no assumptions about the number of word senses.

The sense induction algorithm has three meta-parameters: the ego-network size (N) of the target ego word t ; the ego-network connectivity (n) is the maximum number of connections the neighbour v is allowed to have within the ego-network; the minimum size of the cluster k . The n parameter regulates the granularity of the inventory. In our experiments, we set the N to 200, n to 50, 100 or 200 and k to 5 or 15 to obtain different granulates, cf. (Biemann, 2010).

Each word in a sense cluster has a weight which is equal to the similarity score between this word and the ambiguous word t .

3.4 Pooling of Word Vectors

At this stage, we calculate sense embeddings for each sense in the induced inventory. We assume that a word sense is a composition of words that represent the sense. We define a sense vector as a function of word vectors representing cluster items. Let W be a set of all words in the training corpus and let $S_i = \{w_1, \dots, w_n\} \subseteq W$ be

	TWSI	JBT	w2v
table (furniture)	counter, console, bench, dinner table, dining table, desk, surface, bar, board	chair, room, desk, pulpit, couch, furniture, fireplace, bench, door, box, railing, tray	tray, bottom, bucket, basket, cup, pile, bracket, pot, cue, plate, jar, platter, ladder
table (data)	chart, list, index, graph, columned list, tabulation, standings, diagram, ranking	procedure, system, datum, process, mechanism, tool, method, database, calculation, scheme	diagram, brackets, stack, list, parenthesis, playfield, dropdown, cube, hash, results, tab
table (negotiations)	surface, counter, console, bargaining table, platform, negotiable, machine plate, level	—	—
table (geo)	level, plateau, plain, flatland, saturation level, water table, geographical level, water level	—	—

Table 2: Word sense clusters from inventories derived from the Wikipedia corpus via crowdsourcing (TWSI), JoBimText (JBT) and word embeddings (w2v). The sense labels are introduced for readability.

a sense cluster obtained during the previous step. Consider a function $vec_w : W \rightarrow \mathbb{R}^m$ that maps words to their vectors and a function $\gamma_i : W \rightarrow \mathbb{R}$ that maps cluster words to their weight in the cluster S_i . We experimented with two ways to calculate sense vectors: unweighted average of word vectors:

$$\mathbf{s}_i = \frac{\sum_{k=1}^n vec_w(w_k)}{n};$$

and weighted average of word vectors:

$$\mathbf{s}_i = \frac{\sum_{k=1}^n \gamma_i(w_k) vec_w(w_k)}{\sum_{k=1}^n \gamma_i(w_k)}.$$

Table 1 provides an example of weighted pooling results. While the original neighbours of the word “table” contain words related to both furniture and data, the neighbours of the sense vectors are either related to furniture or data, but not to both at the same time. Besides, each neighbour of a sense vector has a sense identifier as we calculate cosine between sense vectors, not word vectors.

4 Word Sense Disambiguation

This section describes how sense vectors are used to disambiguate a word in a context.

Given a target word w and its context words $C = \{c_1, \dots, c_k\}$, we first map w to a set of its sense vectors according to the inventory: $S = \{\mathbf{s}_1, \dots, \mathbf{s}_n\}$. We use two strategies to choose a correct sense taking vectors for context words either from the matrix of context embeddings or from the matrix of word vectors. The first one is based on sense probability in given context:

$$s^* = \arg \max_i P(C|\mathbf{s}_i) = \arg \max_i \frac{1}{1 + e^{-\bar{c}_c \cdot \mathbf{s}_i}},$$

where \bar{c}_c is the mean of context embeddings: $k^{-1} \sum_{i=1}^k vec_c(c_i)$ and functions $vec_c : W \rightarrow \mathbb{R}^m$ map context words to context embeddings. Using the mean of context embeddings to calculate sense probability is natural with the CBOW because this model optimises exactly the same mean to have high scalar product with word embeddings for words occurred in context and low scalar product for random words (Mikolov et al., 2013).

The second disambiguation strategy is based on similarity between sense and context:

$$s^* = \arg \max_i sim(\mathbf{s}_i, C) = \arg \max_i \frac{\bar{c}_w \cdot \mathbf{s}_i}{\|\bar{c}_w\| \cdot \|\mathbf{s}_i\|},$$

where \bar{c}_w is the mean of word embeddings: $k^{-1} \sum_{i=1}^k vec_w(c_i)$. The latter method uses only word vectors (vec_w) and require no context vectors (vec_c). This is practical, as the standard implementation of *word2vec* does not save context embeddings and thus most pre-computed models provide only word vectors.

To improve WSD performance we also apply context filtering. Typically, only several words in context are relevant for sense disambiguation, like “chairs” and “kitchen” are for “table” in “They bought a table and chairs for kitchen.” For each word c_j in context $C = \{c_1, \dots, c_k\}$ we calculate a score that quantifies how well it discriminates the senses:

$$\max_i f(\mathbf{s}_i, c_j) - \min_i f(\mathbf{s}_i, c_j),$$

where \mathbf{s}_i iterates over senses of the ambiguous word and f is one of our disambiguation strategies: either $P(c_j|\mathbf{s}_i)$ or $sim(\mathbf{s}_i, c_j)$. The p most discriminative context words are used for disambiguation.

	Full TWSI		Balanced TWSI	
	w2v	GBT	w2v	GBT
no filter	0.676	0.669	0.383	0.397
filter, $p = 5$	0.679	0.674	0.386	0.403
filter, $p = 3$	0.681	0.676	0.387	0.409
filter, $p = 2$	0.683	0.678	0.389	0.410
filter, $p = 1$	0.683	0.676	0.390	0.410

Table 4: Influence of context filtering on disambiguation in terms of F-score. The models were trained on Wikipedia corpus; the w2v is based on weighted pooling and similarity-based disambiguation. All differences between filtered and unfiltered models are significant ($p < 0.05$).

5 Experiments

We evaluate our method on two complementary datasets: (1) a crowdsourced collection of sense-labeled contexts; and (2) a commonly used SemEval dataset.

5.1 Evaluation on TWSI

The goal of this evaluation is to test different configurations of our approach on a large-scale dataset, i.e. it is used for development purposes.

Dataset. This test collection is based on a large-scale crowdsourced resource by Biemann (2012) that comprises 1,012 frequent nouns with average polysemy of 2.26 senses per word. For these nouns the dataset provides 145,140 annotated sentences sampled from Wikipedia. Besides, it is accompanied by an explicit sense inventory, where each sense is represented with a list of words that can substitute target noun in a given sentence.

The sense distribution across sentences in the dataset is skewed, resulting in 79% of contexts assigned to the most frequent senses. Therefore, in addition to the full TWSI dataset, we also use a balanced subset that has no bias towards the Most Frequent Sense (MFS). This dataset features 6,165 contexts with five contexts per sense excluding monosemous words.

Evaluation metrics. To compute WSD performance, we create an explicit mapping between the system-provided sense inventory and the TWSI senses: senses are represented as bag of words vectors, which are compared using cosine similarity. Every induced sense gets assigned to at most one TWSI sense. Once the mapping is completed, we can calculate precision and recall of sense prediction with respect to the original TWSI labeling.

Performance of a disambiguation model depends on quality of the sense mapping. These baselines facilitate interpretation of results:

- **Upper bound of the induced inventory** selects the correct sense for the context, but only if the mapping exist for this sense.
- **MFS of the TWSI inventory** assigns the most frequent sense in the TWSI dataset.
- **MFS of the induced inventory** assigns the identifier of the largest sense cluster.
- **Random sense baseline** of the TWSI and induced sense inventories.

Discussion of results. Table 2 presents examples of the senses induced via clustering of nearest neighbours generated by word embeddings (w2v) and JBT as compared to the inventory produced via crowdsourcing (TWSI). The TWSI contains more senses (2.26 on average), while induced ones have less senses (1.56 and 1.64, respectively). The senses in the table are arranged in the way they are mapped to TWSI during evaluation.

Table 3 illustrates how the granularity of the inventory influences WSD performance. The more granular the sense inventory, the better the match between the TWSI and the induced inventory can be established (mind that we map every induced sense to at most one TWSI sense). Therefore, the upper bound of WSD performance is maximal for the most fine-grained inventories.

However, the relation of actual WSD performance to granularity is inverse: the lower the number of senses, the higher the WSD performance (in the limit, we converge to the strong MFS baseline). We select a coarse-grained inventory for our further experiments ($n=200$, $k = 15$).

Table 4 illustrates the fact that using context filtering positively impacts disambiguation performance, reaching optimal characteristics when two context words are used.

Finally, Figure 3 presents results of our experiments on the full and sense-balanced TWSI datasets. First of all, our models significantly outperform random sense baseline of both TWSI and induced inventories. Secondly, we observe that pooling vectors using similarity scores as weights is better than unweighted pooling. Indeed, some clusters may contain irrelevant words and thus their contribution should be discounted. Third, we observe that using similarity-based disambiguation mechanism yields better results as compared

Inventory	#Senses	Upper-bound of Inventory			Probability-based WSD		
		Prec.	Recall	F-score	Prec.	Recall	F-score
TWSI	2.26	1.000	1.000	1.000	0.484	0.483	0.484
w2v wiki, $k = 15$	1.56	1.000	0.479	0.648	0.367	0.366	0.366
GBT wiki, $n = 200, k = 15$	1.64	1.000	0.488	0.656	0.383	0.383	0.383
GBT ukWaC, $n = 200, k = 15$	1.89	1.000	0.526	0.690	0.360	0.360	0.360
GBT wiki, $n = 200, k = 5$	2.55	1.000	0.598	0.748	0.338	0.338	0.338
GBT wiki, $n = 100, k = 5$	3.59	1.000	0.671	0.803	0.305	0.305	0.305
GBT wiki, $n = 50, k = 5$	5.13	1.000	0.724	0.840	0.275	0.275	0.275

Table 3: Upper-bound and actual value of the WSD performance on the sense-balanced TWSI dataset, function of sense inventory used for unweighted pooling of word vectors.

□ random ■ mean -- prob. ▨ weighted – prob. ▨ weighted -- sim. ■ weighted -- sim. -- filter ($p=2$) □ MFS ▨ upper bound

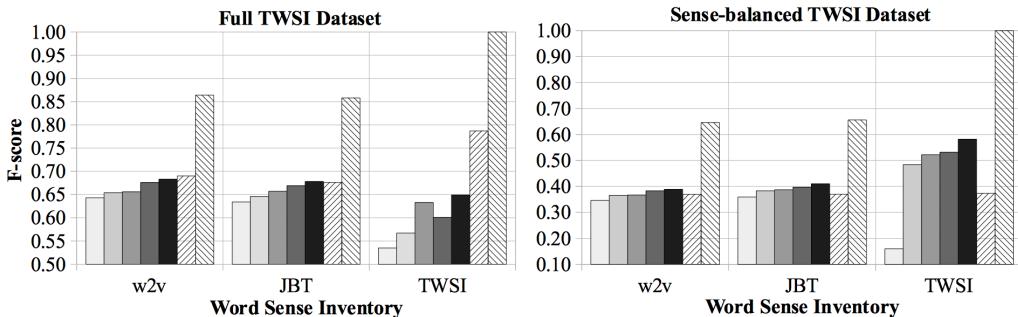


Figure 3: WSD performance of our method trained on the Wikipedia corpus on the full (on the left) and on the sense-balanced (on the right) TWSI dataset. The w2v models are based on the CBOW with 300 dimensions and context window size 3. The JBT models are computed using the Malt parser.

to the mechanism based on probabilities. Indeed, cosine similarity between embeddings proved to be useful for semantic relatedness, yielding state-of-the-art results (Baroni et al., 2014), while there is less evidence about successful use-cases of the CBOW as a language model.

Fourth, we confirm our observation that filtering context words positively impacts WSD performance. Finally, we note that models based on JBT- and w2v-induced sense inventories yield comparable results. However, the JBT inventory shows higher performance (0.410 vs 0.390) on the balanced TWSI, indicating the importance of a precise sense inventory. Finally, using the "gold" TWSI inventory significantly improves the performance on the balanced dataset outperforming models based on induced inventories.

5.2 Evaluation on SemEval-2013 Task 13

The goal of this evaluation is to compare the performance of our method to state-of-the-art unsupervised WSD systems.

Dataset. The SemEval-2013 task 13 "Word Sense Induction for Graded and Non-Graded Senses" (Jurgens and Klaptis, 2013) provides 20 nouns, 20 verbs and 10 adjectives in WordNet-sense-tagged contexts. It contains 20-100 contexts per word, and 4,664 contexts in total, which were drawn from the Open American National Corpus. Participants were asked to cluster these 4,664 instances into groups, with each group corresponding to a distinct word sense.

Evaluation metrics. Performance is measured with three measures that require a mapping of sense inventories (Jaccard Index, Tau and WNDG) and two cluster comparison measures (Fuzzy NMI and Fuzzy B-Cubed).

Discussion of results. We compare our approach to SemEval participants and the AdaGram sense embeddings. The *AI-KU* system (Baskaya et al., 2013) directly clusters test contexts using the k -means algorithm based on lexical substitution features. The *Unimelb* system (Lau et al., 2013) uses a hierarchical topic model to induce and dis-

Model		Supervised Evaluation			Clustering Evaluation	
		Jacc.	Ind.	Tau	WNDCG	FNMI
Baselines	One sense for all	0.171	0.627	0.302	0.000	0.631
	One sense per instance	0.000	0.953	0.000	0.072	0.000
	Most Frequent Sense (MFS)	0.579	0.583	0.431	—	—
SemEval	AI-KU (add1000)	0.176	0.609	0.205	0.033	0.317
	AI-KU	0.176	0.619	0.393	0.066	0.382
	AI-KU (remove5-add1000)	0.228	0.654	0.330	0.040	0.463
	Unimelb (5p)	0.198	0.623	0.374	0.056	0.475
	Unimelb (50k)	0.198	0.633	0.384	0.060	0.494
	UoS (#WN senses)	0.171	0.600	0.298	0.046	0.186
	UoS (top-3)	0.220	0.637	0.370	0.044	0.451
	La Sapienza (1)	0.131	0.544	0.332	—	—
	La Sapienza (2)	0.131	0.535	0.394	—	—
Sense emb.	AdaGram, $\alpha = 0.05$, 100 dim. vectors	0.274	0.644	0.318	0.058	0.470
Our models	w2v – weighted – sim. – filter ($p = 2$)	0.197	0.615	0.291	0.011	0.615
	w2v – weighted – sim. – filter ($p = 2$): nouns	0.179	0.626	0.304	0.011	0.623
	GBT – weighted – sim. – filter ($p = 2$)	0.205	0.624	0.291	0.017	0.598
	GBT – weighted – sim. – filter ($p = 2$): nouns	0.198	0.643	0.310	0.031	0.595
	TWSI – weighted – sim. – filter ($p = 2$): nouns	0.215	0.651	0.318	0.030	0.573

Table 5: The best configurations of our method selected on the TWSI dataset on the SemEval 2013 Task 13 dataset. The w2v-based methods rely on the CBOW model with 100 dimensions and context window size 3. The JBT similarities were computed using the Malt parser. All systems were trained on the ukWaC corpus.

ambiguate word senses. The *UoS* system (Hope and Keller, 2013) is most similar to our approach: to induce senses it builds an ego-network of a word using dependency relations, which is subsequently clustered using a simple graph clustering algorithm. The *La Sapienza* system (Agirre and Soroa, 2009), relies on WordNet to get word senses and perform disambiguation.

Table 5 shows a comparative evaluation of our method on the SemEval dataset. Like above, dependency-based (GBT) word similarities yield slightly better results than word embedding similarity (w2v) for inventory induction. In addition to these two configurations, we also built a model based on the TWSI sense inventory (only for nouns as the TWSI contains nouns only). This model significantly outperforms both GBT- and w2v-based models, thus precise sense inventories greatly influence WSD performance.

As one may observe, performance of the best configurations of our method is comparable to the top-ranked SemEval participants, but is not systematically exceeding their results. AdaGram sometimes outperforms our method, sometimes it is on par, depending on the metric. We interpret these results as an indication of comparability of our method to state-of-the-art approaches.

Finally, note that none of the unsupervised WSD methods discussed in this paper, includ-

ing the top-ranked SemEval submissions and AdaGram, were able to beat the most frequent sense baselines of the respective datasets (with the exception of the balanced version of TWSI). Similar results are observed for other unsupervised WSD methods (Nieto Piña and Johansson, 2016).

6 Conclusion

We presented a novel approach for learning of multi-prototype word embeddings. In contrast to existing approaches that learn sense embeddings directly from the corpus, our approach can operate on existing word embeddings. It can either induce or reuse a word sense inventory. Experiments on two datasets, including a SemEval challenge on word sense induction and disambiguation, show that our approach performs comparably to the state of the art.

An implementation of our method with several pre-trained models is available online.³

Acknowledgments

We acknowledge the support of the Deutsche Forschungsgemeinschaft (DFG) foundation under the project ”JOIN-T: Joining Ontologies and Semantics Induced from Text”.

³<https://github.com/tudarmstadt-lt/sensemgram>

References

- Eneko Agirre and Philip Edmonds. 2007. *Word sense disambiguation: Algorithms and applications*, volume 33. Springer Science & Business Media.
- Eneko Agirre and Aitor Soroa. 2009. Personalizing PageRank for Word Sense Disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the ACL*, pages 33–41, Athens, Greece.
- Satanjeev Banerjee and Ted Pedersen. 2002. An adapted Lesk algorithm for word sense disambiguation using WordNet. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics*, pages 136–145, Mexico City, Mexico.
- Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.
- Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don’t count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 238–247, Baltimore, Maryland. Association for Computational Linguistics.
- Sergey Bartunov, Dmitry Kondrashkin, Anton Osokin, and Dmitry Vetrov. 2016. Breaking sticks and ambiguities with adaptive skip-gram. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*.
- Osman Baskaya, Enis Sert, Volkan Cirik, and Deniz Yuret. 2013. AI-KU: Using Substitute Vectors and Co-Occurrence Modeling for Word Sense Induction and Disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM): SemEval 2013*, volume 2, pages 300–306, Atlanta, Georgia, USA.
- Chris Biemann and Martin Riedl. 2013. Text: Now in 2D! a framework for lexical expansion with contextual similarity. *Journal of Language Modelling*, 1(1):55–95.
- Chris Biemann. 2006. Chinese Whispers: An Efficient Graph Clustering Algorithm and Its Application to Natural Language Processing Problems. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*, pages 73–80, New York City, USA.
- Chris Biemann. 2010. Co-occurrence cluster features for lexical substitutions in context. In *Proceedings of the 5th Workshop on TextGraphs in conjunction with ACL 2010*, Uppsala, Sweden.
- Chris Biemann. 2012. Turk Bootstrap Word Sense Inventory 2.0: A Large-Scale Resource for Lexical Substitution. In *Proceedings of the 8th International Conference on Language Resources and Evaluation*, pages 4038–4042, Istanbul, Turkey.
- Antoine Bordes, Weston Jason, Ronan Collobert, and Yoshua Bengio. 2011. Learning structured embeddings of knowledge bases. In *Proc. AAAI*, San Francisco, CA, USA.
- José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. A unified multilingual semantic representation of concepts. In *Proceedings of the Association for Computational Linguistics*, pages 741–751, Beijing, China.
- Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.
- Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukwac, a very large web-derived corpus of english. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google*, pages 47–54.
- David Hope and Bill Keller. 2013. MaxMax: A Graph-based Soft Clustering Algorithm Applied to Word Sense Induction. In *Proceedings of the 14th International Conference on Computational Linguistics and Intelligent Text Processing - Volume Part I*, pages 368–381, Samos, Greece. Springer-Verlag.
- Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the ACL*, pages 873–882, Jeju Island, Korea.
- David Jurgens and Ioannis Klapftis. 2013. Semeval-2013 task 13: Word sense induction for graded and non-graded senses. In *Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 290–299, Atlanta, Georgia, USA.
- Dan Klein, Kristina Toutanova, H. Tolga Ilhan, Sepandar D. Kamvar, and Christopher D. Manning. 2002. Combining Heterogeneous Classifiers for Word-Sense Disambiguation. In *Proceedings of the ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, volume 8, pages 74–80, Philadelphia, PA.
- Jey Han Lau, Paul Cook, and Timothy Baldwin. 2013. unimelb: Topic Modelling-based Word Sense Induction. In *Second Joint Conference on Lexical and Computational Semantics (*SEM): SemEval 2013*, volume 2, pages 307–311, Atlanta, Georgia, USA.
- Yoong Keok Lee and Hwee Tou Ng. 2002. An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In *Proceedings of the Empirical Methods in Natural Language Processing*, volume 10, pages 41–48, Philadelphia, PA.

- Michael Lesk. 1986. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In *Proceedings of the 5th International Conference on Systems Documentation*, pages 24–26, Toronto, ON, Canada. ACM.
- Jiwei Li and Dan Jurafsky. 2015. Do multi-sense embeddings improve natural language understanding? In *Conference on Empirical Methods in Natural Language Processing, EMNLP'2015*, pages 1722–1732, Lisboa, Portugal.
- Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of ICML*, volume 98, pages 296–304, Madison, WI, USA.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Workshop at International Conference on Learning Representations (ICLR)*, pages 1310–1318, Scottsdale, AZ, USA.
- Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244.
- Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10.
- Arvind Neelakantan, Jeevan Shankar, Alexandre Pasos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069, Doha, Qatar.
- Luis Nieto Piña and Richard Johansson. 2015. A simple and efficient method to generate word sense representations. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, Hissar, Bulgaria, September.
- Luis Nieto Piña and Richard Johansson. 2016. Embedding senses for efficient graph-based word sense disambiguation. In *Proceedings of TextGraphs-10, Proceedings of the Human Language Technology Conference of the NAACL*, San Diego, United States.
- Alexander Panchenko. 2013. *Similarity measures for semantic relation extraction*. Ph.D. thesis, Université catholique de Louvain, Louvain-la-Neuve, Belgium.
- Patrick Pantel and Dekang Lin. 2002. Discovering word senses from text. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 613–619. ACM.
- Ted Pedersen and Rebecca Bruce. 1997. Distinguishing word senses in untagged text. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 197–207, Providence, RI.
- Ted Pedersen, Satanjeev Banerjee, and Siddharth Patwardhan. 2005. Maximizing semantic relatedness to perform word sense disambiguation. *University of Minnesota supercomputing institute research report UMSI*, 25:2005.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar.
- Joseph Reisinger and Raymond J. Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117, Los Angeles, California.
- Sascha Rothe and Hinrich Schütze. 2015. Autoextend: Extending word embeddings to embeddings for synsets and lexemes. In *Proceedings of the Association for Computational Linguistics*, pages 1793–1803, Beijing, China. Association for Computational Linguistics.
- Eugen Ruppert, Jonas Klesy, Martin Riedl, and Chris Biemann. 2015. Rule-based dependency parse collapsing and propagation for german and english. In *Proceedings of the GSCL 2015*, pages 58–66, Duisburg, Germany.
- Gerard Salton, Anita Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.
- Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational linguistics*, 24(1):97–123.
- Fei Tian, Hanjun Dai, Jiang Bian, Bin Gao, Rui Zhang, Enhong Chen, and Tie-Yan Liu. 2014. A probabilistic model for learning multi-prototype word embeddings. In *COLING*, pages 151–160, Dublin, Ireland.
- Jean Véronis. 2004. HyperLex: Lexical cartography for information retrieval. *Computer Speech and Language*, 18:223–252.
- Dominic Widdows and Beate Dorow. 2002. A graph model for unsupervised lexical acquisition. In *Proceedings of the 19th international conference on Computational linguistics*, pages 1–7, Taipei, Taiwan.
- Ziqi Zhang, Anna Lisa Gentile, and Fabio Ciravegna. 2013. Recent advances in methods of lexical semantic relatedness—a survey. *Natural Language Engineering*, 19(04):411–479.

A.12 Paper “Unsupervised Does Not Mean Uninterpretable: The Case for Word Sense Induction and Disambiguation”

A. Panchenko, F. Marten, E. Ruppert, S. Faralli, D. Ustalov, S. P. Ponzetto, and C. Biemann, “Unsupervised, Knowledge-Free, and Interpretable Word Sense Disambiguation,” in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, (Copenhagen, Denmark), pp. 91–96, Association for Computational Linguistics, Sept. 2017

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/E17-1009>

Unsupervised Does Not Mean Uninterpretable: The Case for Word Sense Induction and Disambiguation

Alexander Panchenko[‡], Eugen Ruppert[‡], Stefano Faralli[†],
Simone Paolo Ponzetto[†] and Chris Biemann[‡]

[‡]Language Technology Group, Computer Science Dept., University of Hamburg, Germany

[†]Web and Data Science Group, Computer Science Dept., University of Mannheim, Germany

{panchenko, ruppert, biemann}@informatik.uni-hamburg.de

{faralli, simone}@informatik.uni-mannheim.de

Abstract

The current trend in NLP is the use of highly opaque models, e.g. neural networks and word embeddings. While these models yield state-of-the-art results on a range of tasks, their drawback is poor interpretability. On the example of word sense induction and disambiguation (WSID), we show that it is possible to develop an interpretable model that matches the state-of-the-art models in accuracy. Namely, we present an unsupervised, knowledge-free WSID approach, which is interpretable at three levels: word sense inventory, sense feature representations, and disambiguation procedure. Experiments show that our model performs on par with state-of-the-art word sense embeddings and other unsupervised systems while offering the possibility to justify its decisions in human-readable form.

1 Introduction

A word sense disambiguation (WSD) system takes as input a target word t and its context C . The system returns an identifier of a word sense s_i from the word sense inventory $\{s_1, \dots, s_n\}$ of t , where the senses are typically defined manually in advance. Despite significant progress in methodology during the two last decades (Ide and Véronis, 1998; Agirre and Edmonds, 2007; Moro andNavigli, 2015), WSD is still not widespread in applications (Navigli, 2009), which indicates the need for further progress. The difficulty of the problem largely stems from the lack of domain-specific training data. A fixed sense inventory, such as the one of WordNet (Miller, 1995), may contain irrelevant senses for the given application and at the same time lack relevant domain-specific senses.

Word sense induction from domain-specific corpora is a supposed to solve this problem. However, most approaches to word sense induction and disambiguation, e.g. (Schütze, 1998; Li and Jurafsky, 2015; Bartunov et al., 2016), rely on clustering methods and dense vector representations that make a WSD model uninterpretable as compared to knowledge-based WSD methods.

Interpretability of a statistical model is important as it lets us understand the reasons behind its predictions (Vellido et al., 2011; Freitas, 2014; Li et al., 2016). Interpretability of WSD models (1) lets a user understand why in the given context one observed a given sense (e.g., for educational applications); (2) performs a comprehensive analysis of correct and erroneous predictions, giving rise to improved disambiguation models.

The contribution of this paper is an interpretable unsupervised knowledge-free WSD method. The novelty of our method is in (1) a technique to disambiguation that relies on induced inventories as a pivot for learning sense feature representations, (2) a technique for making induced sense representations interpretable by labeling them with hypernyms and images.

Our method tackles the interpretability issue of the prior methods; it is interpretable at the levels of (1) sense inventory, (2) sense feature representation, and (3) disambiguation procedure. In contrast to word sense induction by context clustering (Schütze (1998), *inter alia*), our method constructs an explicit word sense inventory. The method yields performance comparable to the state-of-the-art unsupervised systems, including two methods based on word sense embeddings. An open source implementation of the method featuring a live demo of several pre-trained models is available online.¹

¹<http://www.jobimtext.org/wsd>

2 Related Work

Multiple designs of WSD systems were proposed (Agirre and Edmonds, 2007; Navigli, 2009). They vary according to the level of supervision and the amount of external knowledge used. Most current systems either make use of lexical resources and/or rely on an explicitly annotated sense corpus.

Supervised approaches use a sense-labeled corpus to train a model, usually building one submodel per target word (Ng, 1997; Lee and Ng, 2002; Klein et al., 2002; Wee, 2010). The IMS system by Zhong and Ng (2010) provides an implementation of the supervised approach to WSD that yields state-of-the-art results. While supervised approaches demonstrate top performance in competitions, they require large amounts of sense-labeled examples per target word.

Knowledge-based approaches rely on a lexical resource that provides a sense inventory and features for disambiguation and vary from the classical Lesk (1986) algorithm that uses word definitions to the Babelfy (Moro et al., 2014) system that uses harnesses a multilingual lexical-semantic network. Classical examples of such approaches include (Banerjee and Pedersen, 2002; Pedersen et al., 2005; Miller et al., 2012). More recently, several methods were proposed to learn sense embeddings on the basis of the sense inventory of a lexical resource (Chen et al., 2014; Rothe and Schütze, 2015; Camacho-Collados et al., 2015; Iacobacci et al., 2015; Nieto Piña and Johansson, 2016).

Unsupervised knowledge-free approaches use neither handcrafted lexical resources nor hand-annotated sense-labeled corpora. Instead, they induce word sense inventories automatically from corpora. Unsupervised WSD methods fall into two main categories: context clustering and word ego-network clustering.

Context clustering approaches, e.g. (Pedersen and Bruce, 1997; Schütze, 1998), represent an instance usually by a vector that characterizes its context, where the definition of context can vary greatly. These vectors of each instance are then clustered. Multi-prototype extensions of the skip-gram model (Mikolov et al., 2013) that use no pre-defined sense inventory learn one embedding word vector per one word sense and are commonly fitted with a disambiguation mechanism (Huang et al., 2012; Tian et al., 2014; Neelakantan et al.,

2014; Bartunov et al., 2016; Li and Jurafsky, 2015; Pelevina et al., 2016). Comparisons of the *AdaGram* (Bartunov et al., 2016) to (Neelakantan et al., 2014) on three SemEval word sense induction and disambiguation datasets show the advantage of the former. For this reason, we use *AdaGram* as a representative of the state-of-the-art word sense embeddings in our experiments. In addition, we compare to SenseGram, an alternative sense embedding based approach by Pelevina et al. (2016). What makes the comparison to the later method interesting is that this approach is similar to ours, but instead of sparse representations the authors rely on word embeddings, making their approach less interpretable.

Word ego-network clustering methods (Lin, 1998; Pantel and Lin, 2002; Widdows and Dorow, 2002; Biemann, 2006; Hope and Keller, 2013) cluster graphs of words semantically related to the ambiguous word. An ego network consists of a single node (ego) together with the nodes they are connected to (alters) and all the edges among those alters (Everett and Borgatti, 2005). In our case, such a network is a local neighborhood of one word. Nodes of the ego-network can be (1) words semantically similar to the target word, as in our approach, or (2) context words relevant to the target, as in the *UoS* system (Hope and Keller, 2013). Graph edges represent semantic relations between words derived using corpus-based methods (e.g. distributional semantics) or gathered from dictionaries. The sense induction process using word graphs is explored by (Widdows and Dorow, 2002; Biemann, 2006; Hope and Keller, 2013). Disambiguation of instances is performed by assigning the sense with the highest overlap between the instance's context words and the words of the sense cluster. Véronis (2004) compiles a corpus with contexts of polysemous nouns using a search engine. A word graph is built by drawing edges between co-occurring words in the gathered corpus, where edges below a certain similarity threshold were discarded. His HyperLex algorithm detects hubs of this graph, which are interpreted as word senses. Disambiguation in this experiment is performed by computing the distance between context words and hubs in this graph.

Di Marco and Navigli (2013) presents a comprehensive study of several graph-based WSI methods including Chinese Whispers, HyperLex, curvature clustering (Dorow et al., 2005). Besides,

authors propose two novel algorithms: Balanced Maximum Spanning Tree Clustering and Squares (B-MST), Triangles and Diamonds (SquaT++). To construct graphs, authors use first-order and second-order relations extracted from a background corpus as well as keywords from snippets. This research goes beyond intrinsic evaluations of induced senses and measures impact of the WSI in the context of an information retrieval via clustering and diversifying Web search results. Depending on the dataset, HyperLex, B-MST or Chinese-Whispers provided the best results.

Our system combines several of above ideas and adds features ensuring interpretability. Most notably, we use a word sense inventory based on clustering word similarities (Pantel and Lin, 2002); for disambiguation we rely on syntactic context features, co-occurrences (Hope and Keller, 2013) and language models (Yuret, 2012).

Interpretable approaches. The need in methods that interpret results of opaque statistical models is widely recognised (Vellido et al., 2011; Vellido et al., 2012; Freitas, 2014; Li et al., 2016; Park et al., 2016). An interpretable WSD system is expected to provide (1) a human-readable sense inventory, (2) human-readable reasons why in a given context c a given sense s_i was detected. Lexical resources, such as WordNet, solve the first problem by providing manually-crafted definitions of senses, examples of usage, hypernyms, and synonyms. The BabelNet (Navigli and Ponzetto, 2010) integrates all these sense representations, adding to them links to external resources, such as Wikipedia, topical category labels, and images representing the sense. The unsupervised models listed above do not feature any of these representations making them much less interpretable as compared to the knowledge-based models. Ruppert et al. (2015) proposed a system for visualising sense inventories derived in an unsupervised way using graph-based distributional semantics. Panchenko (2016) proposed a method for making sense inventory of word sense embeddings interpretable by mapping it to BabelNet.

Our approach was inspired by the knowledge-based system Babelfy (Moro et al., 2014). While the inventory of Babelfy is interpretable as it relies on BabelNet, the system provides no underlying reasons behind sense predictions. Our objective was to reach interpretability level of knowledge-based models within an unsupervised framework.

3 Method: Unsupervised Interpretable Word Sense Disambiguation

Our unsupervised word sense disambiguation method consist of the five steps illustrated in Figure 1: extraction of context features (Section 3.1); computing word and feature similarities (Section 3.2); word sense induction (Section 3.3); labeling of clusters with hypernyms and images (Section 3.4), disambiguation of words in context based on the induced inventory (Section 3.5), and finally interpretation of the model (Section 3.6). Feature similarity and co-occurrence computation steps (drawn with a dashed lines) are optional, since they did not consistently improve performance.

3.1 Extraction of Context Features

The goal of this step is to extract word-feature counts from the input corpus. In particular, we extract three types of features:

Dependency Features. These feature represents a word by a syntactic dependency such as “nn(•,writing)” or “prep.at(sit,•)”, extracted from the Stanford Dependencies (De Marneffe et al., 2006) obtained with the the PCFG model of the Stanford parser (Klein and Manning, 2003). Weights are computed using the Local Mutual Information (LMI) (Evert, 2005). One word is represented with 1000 most significant features.

Co-occurrence Features. This type of features represents a word by another word. We extract the list of words that significantly co-occur in a sentence with the target word in the input corpus based on the log-likelihood as word-feature weight (Dunning, 1993).

Language Model Feature. This type of features are based on a trigram model with Kneser-Ney smoothing (Kneser and Ney, 1995). In particular, a word is represented by (1) right and left context words, e.g. “office_•_and”, (2) two preceding words “new_office_•”, and (3) two succeeding words, e.g. “•_and_chairs”. We use the conditional probabilities of the resulting trigrams as word-feature weights.

3.2 Computing Word and Feature Similarities

The goal of this step is to build a graph of word similarities, such as (table, chair, 0.78). We used the *JoBimText* framework (Biemann and Riedl,

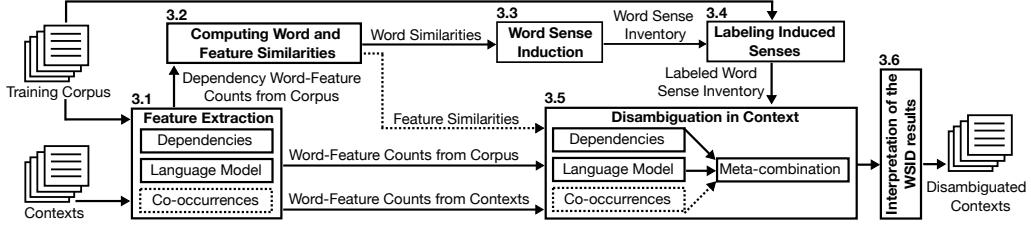


Figure 1: Outline of our unsupervised interpretable method for word sense induction and disambiguation.

2013) as it yields comparable performance on semantic similarity to state-of-the-art dense representations (Mikolov et al., 2013) compared on the WordNet as gold standard (Riedl, 2016), but is interpretable as words are represented by sparse interpretable features. Namely we use dependency-based features as, according to prior evaluations, this kind of features provides state-of-the-art semantic relatedness scores (Padó and Lapata, 2007; Van de Cruys, 2010; Panchenko and Morozova, 2012; Levy and Goldberg, 2014).

First, features of each word are ranked using the LMI metric (Evert, 2005). Second, the word representations are pruned keeping 1000 most salient features per word and 1000 most salient words per feature. The pruning reduces computational complexity and noise. Finally, word similarities are computed as a number of common features for two words. This is again followed by a pruning step in which only the 200 most similar terms are kept to every word. The resulting word similarities are browsable online.²

Note that while words can be characterized with distributions over features, features can vice versa be characterized by a distribution over words. We use this duality to compute feature similarities using the same mechanism and explore their use in disambiguation below.

3.3 Word Sense Induction

We induce a sense inventory by clustering of ego-network of similar words. In our case, an inventory represents senses by a word cluster, such as “chair, bed, bench, stool, sofa, desk, cabinet” for the “furniture” sense of the word “table”.

The sense induction processes one word t of the distributional thesaurus T per iteration. First, we retrieve nodes of the ego-network G of t being the N most similar words of t according to T (see

²Select the “JoBimViz” demo and then the “Stanford (English)” model: <http://www.jobimtext.org>.

Figure 2 (1)). Note that the target word t itself is not part of the ego-network. Second, we connect each node in G to its n most similar words according to T . Finally, the ego-network is clustered with Chinese Whispers (Biemann, 2006), a non-parametric algorithm that discovers the number of senses automatically. The n parameter regulates the granularity of the inventory: we experiment with $n \in \{200, 100, 50\}$ and $N = 200$.

The choice of Chinese Whispers among other algorithms, such as HyperLex (Véronis, 2004) or MCL (Van Dongen, 2008), was motivated by the absence of meta-parameters and its comparable performance on the WSI task to the state-of-the-art (Di Marco and Navigli, 2013).

3.4 Labeling Induced Senses with Hyponyms and Images

Each sense cluster is automatically labeled to improve its interpretability. First, we extract hyponyms from the input corpus using Hearst (1992) patterns. Second, we rank hyponyms relevant to the cluster by a product of two scores: the *hyponym relevance* score, calculated as $\sum_{w \in \text{cluster}} \text{sim}(t, w) \text{freq}(w, h)$, and the *hyponym coverage* score, calculated as $\sum_{w \in \text{cluster}} \min(\text{freq}(w, h), 1)$. Here the $\text{sim}(t, w)$ is the relatedness of the cluster word w to the target word t , and the $\text{freq}(w, h)$ is the frequency of the hyponymy relation (w, h) as extracted via patterns. Thus, a high-ranked hyponym h has high relevance, but also is confirmed by several cluster words. This stage results in a ranked list of labels that specify the word sense, for which we here show the first one, e.g. “table (furniture)” or “table (data)”.

Faralli and Navigli (2012) showed that web search engines can be used to bootstrap sense-related information. To further improve interpretability of induced senses, we assign an image to each word in the cluster (see Figure 2) by query-

ing the Bing image search API³ using the query composed of the target word and its hypernym, e.g. “jaguar car”. The first hit of this query is selected to represent the induced word sense.

Algorithm 1: Unsupervised WSD of the word t based on the induced word sense inventory I .

```

input : Word  $t$ , context features  $C$ , sense inventory  $I$ ,
        word-feature table  $F$ , use largest cluster
        back-off  $LCB$ , use feature expansion  $FE$ .
output: Sense of the target word  $t$  in inventory  $I$  and
        confidence score.

1  $S \leftarrow \text{getSenses}(I, t)$ 
2 if  $FE$  then
3   |  $C \leftarrow \text{featureExpansion}(C)$ 
4 end
5 foreach  $(sense, cluster) \in S$  do
6   |  $\alpha[sense] \leftarrow \{\}$ 
7   | foreach  $w \in cluster$  do
8     |   | foreach  $c \in C$  do
9       |     |   |  $\alpha[sense] \leftarrow \alpha[sense] \cup F(w, c)$ 
10      |   | end
11    | end
12 end
13 if  $\max_{sense \in S} \text{mean}(\alpha[sense]) = 0$  then
14   | if  $LCB$  then
15     |   | return  $\arg \max_{(sense, cluster) \in S} |cluster|$ 
16   | else
17     |   | return  $-1 // \text{reject to classify}$ 
18   | end
19 else
20   |   | return  $\arg \max_{(sense, cluster) \in S} \text{mean}(\alpha[sense])$ 
21 end

```

3.5 Word Sense Disambiguation with Induced Word Sense Inventory

To disambiguate a target word t in context, we extract context features C and pass them to Algorithm 1. We use the induced sense inventory I and select the sense that has the largest weighted feature overlap with context features or fall back to the largest cluster back-off when context features C do not match the learned sense representations.

The algorithm starts by retrieving induced sense clusters of the target word (line 1). Next, the method starts to accumulate context feature weights of each *sense* in $\alpha[sense]$. Each word w in a sense *cluster* brings all its word-feature counts $F(w, c)$: see lines 5-12. Finally, a *sense* that maximizes mean weight across all context features is chosen (lines 13-21). Optionally, we can resort to the largest cluster back-off (LCB) strategy in case if no context features match sense representations.

³<https://azure.microsoft.com/en-us/services/cognitive-services/search>

Note that the induced inventory I is used as a pivot to aggregate word-feature counts $F(w, c)$ of the words in the *cluster* in order to build feature representations of each induced *sense*. We assume that the sets of similar words per sense are compatible with each other’s context. Thus, we can aggregate ambiguous feature representations of words in a sense cluster. In a way, occurrences of cluster members form the training set for the sense, i.e. contexts of {chair, bed, bench, stool, sofa, desk, cabinet}, add to the representation of “table (furniture)” in the model. Here, ambiguous cluster members like “chair” (which could also mean “chairman”) add some noise, but its influence is dwarfed by the aggregation over all cluster members. Besides, it is unlikely that the target (“table”) and the cluster member (“chair”) share the same homonymy, thus noisy context features hardly play a role when disambiguating the target in context. For instance, for scoring using language model features, we retrieve the context of the target word and substitute the target word one by one of the cluster words. To close the gap between the aggregated dependency per sense $\alpha[sense]$ and dependencies observed in the target’s context C , we use the similarity of features: we expand every feature $c \in C$ with 200 of most similar features and use them as additional features (lines 2-4).

We run disambiguation independently for each of the feature types listed above, e.g. dependencies or co-occurrences. Next, independent predictions are combined using the majority-voting rule.

3.6 Interpretability of the Method

Results of disambiguation can be interpreted by humans as illustrated by Figure 2. In particular, our approach is interpretable at three levels:

1. Word sense inventory. To make induced word sense inventories interpretable we display senses of each word as an ego-network of its semantically related words. For instance, the network of the word “table” in our example is constructed from two tightly related groups of words that correspond to “furniture” and “data” senses. These labels of the clusters are obtained automatically (see Section 3.4).

While alternative methods, such as *AdaGram*, can generate sense clusters, our approach makes the senses better interpretable due to hypernyms and image labels that summarize senses.

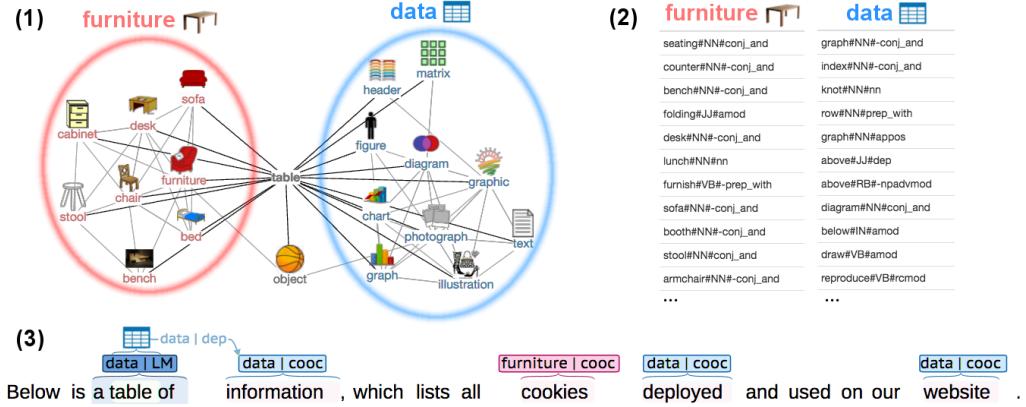


Figure 2: Interpretation of the senses of the word “table” at three levels by our method: (1) word sense inventory; (2) sense feature representation; (3) results of disambiguation in context. The sense labels (“furniture” and “data”) are obtained automatically based on cluster labeling with hypernyms. The images associated with the senses are retrieved using a search engine:“table data” and “table furniture”.

2. Sense feature representation. Each sense in our model is characterized by a list of sparse features ordered by relevance to the sense. Figure 2 (2) shows most salient dependency features to senses of the word “table”. These feature representations are obtained by aggregating features of sense cluster words.

In systems based on dense vector representations, there is no straightforward way to get the most salient features of a sense, which makes the analysis of learned representations problematic.

3. Disambiguation method. To provide the reasons for sense assignment in context, our method highlights the most discriminative context features that caused the prediction. The discriminative power of a feature is defined as the ratio between its weights for different senses.

In Figure 2 (3) words “information”, “cookies”, “deployed” and “website” are highlighted as they are most discriminative and intuitively indicate on the “data” sense of the word “table” as opposed to the “furniture” sense. The same is observed for other types of features. For instance, the syntactic dependency to the word “information” is specific to the “data” sense.

Alternative unsupervised WSD methods that rely on word sense embeddings make it difficult to explain sense assignment in context due to the use of dense features whose dimensions are not interpretable.

4 Experiments

We use two lexical sample collections suitable for evaluation of unsupervised WSD systems. The first one is the Turk Bootstrap Word Sense Inventory (TWSI) dataset introduced by Biemann (2012). It is used for testing different configurations of our approach. The second collection, the SemEval 2013 word sense induction dataset by Jurgens and Klapaftis (2013), is used to compare our approach to existing systems. In both datasets, to measure WSD performance, induced senses are mapped to gold standard senses. In experiments with the TWSI dataset, the models were trained on the Wikipedia corpus⁴ while in experiments with the SemEval datasets models are trained on the ukWaC corpus (Baroni et al., 2009) for a fair comparison with other participants.

4.1 TWSI Dataset

4.1.1 Dataset and Evaluation Metrics

This test collection is based on a crowdsourced resource that comprises 1,012 frequent nouns with 2,333 senses and average polysemy of 2.31 senses per word. For these nouns, 145,140 annotated sentences are provided. Besides, a sense inventory is explicitly provided, where each sense is represented with a list of words that can substitute target noun in a given sentence. The sense distribution across sentences in the dataset is highly

⁴We use a Wikipedia dump from September 2015: <http://doi.org/10.5281/zenodo.229904>

skewed as 79% of contexts are assigned to the most frequent senses. Thus, in addition to the full TWSI dataset, we also use a balanced subset featuring five contexts per sense and 6,166 sentences to assess the quality of the disambiguation mechanism for smaller senses. This dataset contains no monosemous words to completely remove the bias of the most frequent sense. Note that de-biasing the evaluation set does not de-bias the word sense inventory, thus the task becomes harder for the balanced subset.

For the TWSI evaluation, we create an explicit mapping between the system-provided sense inventory and the TWSI word senses: senses are represented as the bag of words, which are compared using cosine similarity. Every induced sense gets assigned at most one TWSI sense. Once the mapping is completed, we calculate Precision, Recall, and F-measure. We use the following baselines to facilitate interpretation of the results: (1) MFS of the TWSI inventory always assigns the most frequent sense in the TWSI dataset; (2) LCB of the induced inventory always assigns the largest sense cluster; (3) Upper bound of the induced vocabulary always selects the correct sense for the context, but only if the mapping exists for this sense; (4) Random sense of the TWSI and the induced inventories.

4.1.2 Discussion of Results

The results of the TWSI evaluation are presented in Table 1. In accordance with prior art in word sense disambiguation, the most frequent sense (MFS) proved to be a strong baseline, reaching an F-score of 0.787, while the random sense over the TWSI inventory drops to 0.536. The upper bound on our induced inventory (F-score of 0.900) shows that the sense mapping technique used prior to evaluation does not drastically distort the evaluation scores. The LCB baseline of the induced inventory achieves an F-score of 0.691, demonstrating the efficiency of the LCB technique.

Let us first consider models based on single features. Dependency features yield the highest precision of 0.728, but have a moderate recall of 0.343 since they rarely match due to their sparsity. The LCB strategy for these rejected contexts helps to improve recall at cost of precision. Co-occurrence features yield significantly lower precision than the dependency-based features, but their recall is higher. Finally, the language model features yield very balanced results in terms of

both precision and recall. Yet, the precision of the model based on this feature type is significantly lower than that of dependencies.

Not all combinations improve results, e.g. combination of three types of features yields inferior results as compared to the language model alone. However, a combination of the language model with dependency features does provide an improvement over the single models as both these models bring strong signal of complementary nature about the semantics of the context. The dependency features represent syntactic information, while the LM features represent lexical information. This improvement is even more pronounced in the case of the balanced TWSI dataset. This combined model yields the best F-scores overall.

Table 2 presents the effect of the feature expansion based on the graph of similar features. For a low-recall model such the one based on syntactic dependencies, feature expansion makes a lot of sense: it almost doubles recall, while losing some precision. The gain in F-score using this technique is almost 20 points on the full TWSI dataset. However, the need for such expansion vanishes when two principally different types of features (precise syntactic dependencies and high-coverage trigram language model) are combined. Both precision and F-score of this combined model outperforms that of the dependency-based model with feature expansion by a large margin.

Figure 3 illustrates how granularity of the induced sense inventory influences WSD performance. For this experiment, we constructed three inventories, setting the number of most similar words in the ego-network n to 200, 100 and 50. These settings produced inventories with respectively 1.96, 2.98 and 5.21 average senses per target word. We observe that a higher sense granularity leads to lower F-scores. This can be explained because of (1) the fact that granularity of the TWSI is similar to granularity of the most coarse-grained inventory; (2) the higher the number of senses, the higher the chance to make a wrong sense assignment; (3) due to the reduced size of individual clusters, we get less signal per sense cluster and noise becomes more pronounced.

To summarize, the best precision is reached by a model based on un-expanded dependencies and the best F-score can be obtained by a combination of models based on un-expanded dependency features and language model features.

Model	#Senses	Full TWSI			Sense-Balanced TWSI		
		Prec.	Recall	F-score	Prec.	Recall	F-score
MFS of the TWSI inventory	2.31	0.787	0.787	0.787	0.373	0.373	0.373
Random Sense of the TWSI inventory	2.31	0.536	0.534	0.535	0.160	0.160	0.160
Upper bound of the induced inventory	1.96	1.000	0.819	0.900	1.000	0.598	0.748
Largest Cluster Back-Off (LCB) of the induced inventory	1.96	0.691	0.690	0.691	0.371	0.371	0.371
Random sense of the induced inventory	1.96	0.559	0.558	0.558	0.325	0.324	0.324
Dependencies	1.96	0.728	0.343	0.466	0.432	0.190	0.263
Dependencies + LCB	1.96	0.689	0.680	0.684	0.388	0.385	0.387
Co-occurrences (Cooc)	1.96	0.570	0.563	0.566	0.336	0.333	0.335
Language Model (LM)	1.96	0.685	0.677	0.681	0.416	0.412	0.414
Dependencies + LM + Cooc	1.96	0.644	0.636	0.640	0.388	0.386	0.387
Dependencies + LM	1.96	0.689	0.681	0.685	0.426	0.422	0.424

Table 1: WSD performance of different configurations of our method on the full and the sense-balanced TWSI datasets based on the coarse inventory with 1.96 senses/word ($N = 200$, $n = 200$).

Model	Precision	Recall	F-score	Precision	Recall	F-score
Dependencies	0.728	0.343	0.466	0.432	0.190	0.263
Dependencies Exp.	0.687	0.633	0.659	0.414	0.379	0.396
Dependencies + LM	0.689	0.681	0.685	0.426	0.422	0.424
Dependencies Exp. + LM	0.684	0.676	0.680	0.412	0.408	0.410

Table 2: Effect of the feature expansion: performance on the full (on the left) and the sense-balanced (on the right) TWSI datasets. The models were trained on the Wikipedia corpus using the coarse inventory (1.96 senses per word). The best results overall are underlined.

4.2 SemEval 2013 Task 13 Dataset

4.2.1 Dataset and Evaluation Metrics

The task of word sense induction for graded and non-graded senses provides 20 nouns, 20 verbs and 10 adjectives in WordNet-sense-tagged contexts. It contains 20-100 contexts per word, and 4,664 contexts in total with 6,73 sense per word in average. Participants were asked to cluster instances into groups corresponding to distinct word senses. Instances with multiple senses were labeled with a score between 0 and 1.

Performance is measured with three measures that require a mapping of inventories (Jaccard Index, Tau, WNDCG) and two cluster comparison measures (Fuzzy NMI, Fuzzy B-Cubed).

4.2.2 Discussion of Results

Table 3 presents results of evaluation of the best configuration of our approach trained on the ukWaC corpus. We compare our approach to four SemEval participants and two state-of-the-art systems based on word sense embeddings: *AdaGram* (Bartunov et al., 2016) based on Bayesian stick-breaking process⁵ and *SenseGram* (Pelevina et al., 2016) based on clustering of ego-network

generated using word embeddings⁶. The *AI-KU* system (Baskaya et al., 2013) directly clusters test contexts using the k -means algorithm based on lexical substitution features. The *Unimelb* system (Lau et al., 2013) uses one hierarchical topic model to induce and disambiguate senses of one word. The *UoS* system (Hope and Keller, 2013) induces senses by building an ego-network of a word using dependency relations, which is subsequently clustered using the MaxMax clustering algorithm. The *La Sapienza* system (Jurgens and Klapaftis, 2013), relies on WordNet for the sense inventory and disambiguation.

In contrast to the TWSI evaluation, the most fine-grained model yields the best scores, yet the inventory of the task is also more fine-grained than the one of the TWSI (7.08 vs. 2.31 avg. senses per word). Our method outperforms the knowledge-based system of *La Sapienza* according to two of three metrics metrics and the *SenseGram* system based on sense embeddings according to four of five metrics. Note that *SenseGram* outperforms all other systems according to the Fuzzy B-Cubed metric, which is maximized in the “All instances, One sense” settings. Thus this result may be due to

⁵<https://github.com/sbos/AdaGram.jl>

⁶<https://github.com/tudarmstadt-lt/sensemgram>

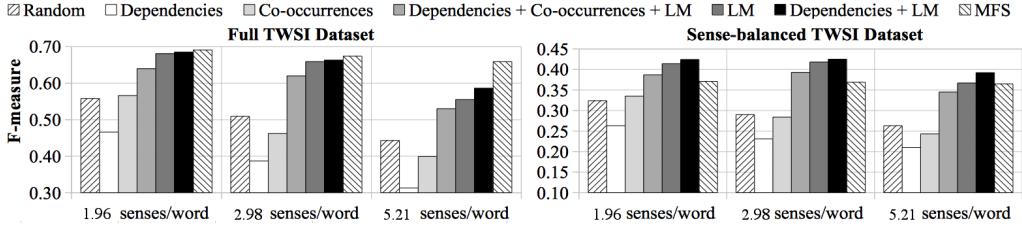


Figure 3: Impact of word sense inventory granularity on WSD performance: the TWSI dataset.

Model	Jacc. Ind.	Tau	WNDCG	Fuzzy NMI	Fuzzy B-Cubed
All Instances, One sense	0.192	0.609	0.288	0.000	0.623
1 sense per instance	0.000	0.953	0.000	0.072	0.000
Most Frequent Sense	0.552	0.560	0.412	–	–
AI-KU	0.197	0.620	0.387	0.065	0.390
AI-KU (remove5-add1000)	0.245	0.642	0.332	0.039	0.451
Unimelb (50k)	0.213	0.620	0.371	0.060	0.483
UoS (top-3)	0.232	0.625	0.374	0.045	0.448
La Sapienza (2)	0.149	0.510	0.383	–	–
AdaGram, $\alpha = 0.05$, 100 dim. vectors	0.274	0.644	0.318	0.058	0.470
SenseGram, 100 dim., CBOW, weight, sim., $p = 2$	0.197	0.615	0.291	0.011	0.615
Dependencies + LM (1.96 senses/word)	0.239	0.634	0.300	0.041	0.513
Dependencies + LM (2.98 senses/word)	0.242	0.634	0.300	0.041	0.504
Dependencies + LM (5.21 senses/word)	0.253	0.638	0.300	0.041	0.479

Table 3: WSD performance of the best configuration of our method identified on the TWSI dataset as compared to participants of the SemEval 2013 Task 13 and two systems based on word sense embeddings (AdaGram and SenseGram). All models were trained on the ukWaC corpus.

difference in granularities: the average polysemy of the *SenseGram* model is 1.56, while the polysemy of our models range from 1.96 to 5.21.

Besides, our system performs comparably to the top unsupervised systems participated in the competition: It is on par with the top SemEval submissions (*AI-KU* and *UoS*) and the another system based on embeddings (*AdaGram*), in terms of four out of five metrics (Jaccard Index, Tau, Fuzzy B-Cubed, Fuzzy NMI).

Therefore, we conclude that our system yields comparable results to the state-of-the-art unsupervised systems. Note, however, that none of the rivaling systems has a comparable level of interpretability to our approach. This is where our method is unique in the class of unsupervised methods: feature representations and disambiguation procedure of the neural-based *AdaGram* and *SenseGram* systems cannot be straightforwardly interpreted. Besides, inventories of the existing systems are represented as ranked lists of words lacking features that improve readability, such as hypernyms and images.

5 Conclusion

In this paper, we have presented a novel method for word sense induction and disambiguation that relies on a meta-combination of dependency features with a language model. The majority of existing unsupervised approaches focus on optimizing the accuracy of the method, sacrificing its interpretability due to the use of opaque models, such as neural networks. In contrast, our approach places a focus on interpretability with the help of sparse readable features. While being interpretable at three levels (sense inventory, sense representations and disambiguation), our method is competitive to the state-of-the-art, including two recent approaches based on sense embeddings, in a word sense induction task. Therefore, it is possible to match the performance of accurate, but opaque methods when interpretability matters.

Acknowledgments

We acknowledge the support of the Deutsche Forschungsgemeinschaft (DFG) foundation under the JOIN-T project.

References

- Eneko Agirre and Philip G. Edmonds. 2007. *Word sense disambiguation: Algorithms and applications*, volume 33. Springer Science & Business Media.
- Satanjeev Banerjee and Ted Pedersen. 2002. An adapted Lesk algorithm for word sense disambiguation using WordNet. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics*, pages 136–145, Mexico City, Mexico. Springer.
- Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3):209–226.
- Sergey Bartunov, Dmitry Kondrashkin, Anton Osokin, and Dmitry Vetrov. 2016. Breaking sticks and ambiguities with adaptive skip-gram. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS'2016)*, Cadiz, Spain.
- Osman Baskaya, Enis Sert, Volkan Cirik, and Deniz Yuret. 2013. AI-KU: Using Substitute Vectors and Co-Occurrence Modeling for Word Sense Induction and Disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 300–306, Atlanta, GA, USA. Association for Computational Linguistics.
- Chris Biemann and Martin Riedl. 2013. Text: Now in 2D! a framework for lexical expansion with contextual similarity. *Journal of Language Modelling*, 1(1):55–95.
- Chris Biemann. 2006. Chinese Whispers: an efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of the first workshop on graph based methods for natural language processing*, pages 73–80, New York City, NY, USA. Association for Computational Linguistics.
- Chris Biemann. 2012. Turk Bootstrap Word Sense Inventory 2.0: A Large-Scale Resource for Lexical Substitution. In *Proceedings of the 8th International Conference on Language Resources and Evaluation*, pages 4038–4042, Istanbul, Turkey. European Language Resources Association.
- José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. NASARI: a novel approach to a semantically-aware representation of items. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 567–577, Denver, CO, USA. Association for Computational Linguistics.
- Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. 2014. A unified model for word sense representation and disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1025–1035, Doha, Qatar. Association for Computational Linguistics.
- Marie-Catherine De Marneffe, Bill MacCartney, Christopher D. Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th Language Resources and Evaluation Conference (LREC'2006)*, pages 449–454, Genova, Italy. European Language Resources Association.
- Antonio Di Marco and Roberto Navigli. 2013. Clustering and diversifying web search results with graph-based word sense induction. *Computational Linguistics*, 39(3):709–754.
- Beate Dorow, Dominic Widdows, Katarina Ling, Jean-Pierre Eckmann, Danilo Sergi, and Elisha Moses. 2005. Using curvature and markov clustering in graphs for lexical acquisition and word sense discrimination. In *Proceedings of the Meaning-2005 Workshop*, Trento, Italy.
- Ted Dunning. 1993. Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics*, 19:61–74.
- Martin Everett and Stephen P. Borgatti. 2005. Ego network betweenness. *Social networks*, 27(1):31–38.
- Stefan Evert. 2005. *The Statistics of Word Cooccurrences Word Pairs and Collocations*. Ph.D. thesis, University of Stuttgart.
- Stefano Faralli and Roberto Navigli. 2012. A new minimally-supervised framework for domain word sense disambiguation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1411–1422, Jeju Island, Korea, July. Association for Computational Linguistics.
- Alex A. Freitas. 2014. Comprehensible classification models: a position paper. *ACM SIGKDD Explorations Newsletter*, 15(1):1–10.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics.
- David Hope and Bill Keller. 2013. MaxMax: A Graph-based Soft Clustering Algorithm Applied to Word Sense Induction. In *Proceedings of the 14th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 368–381, Samos, Greece. Springer.

- Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL'2012)*, pages 873–882, Jeju Island, Korea. Association for Computational Linguistics.
- Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. SensEmbed: learning sense embeddings for word and relational similarity. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL'2015)*, pages 95–105, Beijing, China. Association for Computational Linguistics.
- Nancy Ide and Jean Véronis. 1998. Introduction to the special issue on word sense disambiguation: the state of the art. *Computational linguistics*, 24(1):2–40.
- David Jurgens and Ioannis Klapafitis. 2013. Semeval-2013 Task 13: Word Sense Induction for Graded and Non-graded Senses. In *Proceedings of the 2nd Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval'2013)*, pages 290–299, Montreal, Canada. Association for Computational Linguistics.
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan. Association for Computational Linguistics.
- Dan Klein, Kristina Toutanova, H. Tolga Ilhan, Sepandar D. Kamvar, and Christopher D. Manning. 2002. Combining Heterogeneous Classifiers for Word-Sense Disambiguation. In *Proceedings of the ACL'2002 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, volume 8, pages 74–80, Philadelphia, PA, USA. Association for Computational Linguistics.
- Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP-95)*, volume 1, pages 181–184, Detroit, MI, USA. IEEE.
- Jey Han Lau, Paul Cook, and Timothy Baldwin. 2013. unimelb: Topic Modelling-based Word Sense Induction. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 307–311, Atlanta, GA, USA. Association for Computational Linguistics.
- Yoong Keok Lee and Hwee Tou Ng. 2002. An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'2002)*, volume 10, pages 41–48, Philadelphia, PA, USA. Association for Computational Linguistics.
- Michael Lesk. 1986. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In *Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26, Toronto, ON, Canada. ACM.
- Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, MD, USA. Association for Computational Linguistics.
- Jiwei Li and Dan Jurafsky. 2015. Do multi-sense embeddings improve natural language understanding? In *Conference on Empirical Methods in Natural Language Processing (EMNLP'2015)*, pages 1722–1732, Lisboa, Portugal. Association for Computational Linguistics.
- Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in nlp. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691, San Diego, CA, USA. Association for Computational Linguistics.
- Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning (ICML'1998)*, volume 98, pages 296–304, Madison, WI, USA. Morgan Kaufmann Publishers Inc.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Workshop at International Conference on Learning Representations (ICLR)*, pages 1310–1318, Scottsdale, AZ, USA.
- Tristan Miller, Chris Biemann, Torsten Zesch, and Iryna Gurevych. 2012. Using distributional similarity for lexical expansion in knowledge-based word sense disambiguation. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 1781–1796, Mumbai, India. Association for Computational Linguistics.
- George A. Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.
- Andrea Moro and Roberto Navigli. 2015. SemEval-2015 Task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 288–297, Denver, CO, USA. Association for Computational Linguistics.

- Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244.
- Roberto Navigli and Simone Paolo Ponzetto. 2010. Babelnet: Building a very large multilingual semantic network. In *Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics*, pages 216–225, Uppsala, Sweden.
- Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10.
- Arvind Neelakantan, Jeevan Shankar, Alexandre Pasos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069, Doha, Qatar. Association for Computational Linguistics.
- Hwee Tou Ng. 1997. Exemplar-Based Word Sense Disambiguation: Some Recent Improvements. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 208–213, Providence, RI, USA. Association for Computational Linguistics.
- Luis Nieto Piña and Richard Johansson. 2016. Embedding senses for efficient graph-based word sense disambiguation. In *Proceedings of TextGraphs-10: the Workshop on Graph-based Methods for Natural Language Processing*, pages 1–5, San Diego, CA, USA. Association for Computational Linguistics.
- Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.
- Alexander Panchenko and Olga Morozova. 2012. A study of hybrid similarity measures for semantic relation extraction. In *Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data*, pages 10–18, Avignon, France. Association for Computational Linguistics.
- Alexander Panchenko. 2016. Best of both worlds: Making word sense embeddings interpretable. In *Proceedings of the 10th Language Resources and Evaluation Conference (LREC'2016)*, pages 2649–2655, Portorož, Slovenia. European Language Resources Association (ELRA).
- Patrick Pantel and Dekang Lin. 2002. Discovering word senses from text. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 613–619, Edmonton, AB, Canada.
- Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. 2016. Attentive explanations: Justifying decisions and pointing to the evidence. *arXiv preprint arXiv:1612.04757*.
- Ted Pedersen and Rebecca Bruce. 1997. Distinguishing word senses in untagged text. In *Proceedings of the 2nd Conference on Empirical Methods in Natural Language Processing (EMNLP'1997)*, pages 197–207, Providence, RI, USA. Association for Computational Linguistics.
- Ted Pedersen, Satanjeev Banerjee, and Siddharth Patwardhan. 2005. Maximizing semantic relatedness to perform word sense disambiguation. *University of Minnesota supercomputing institute research report UMSI*, 25:2005.
- Maria Pelevina, Nikolay Arefiev, Chris Biemann, and Alexander Panchenko. 2016. Making sense of word embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 174–183, Berlin, Germany. Association for Computational Linguistics.
- Martin Riedl. 2016. *Unsupervised Methods for Learning and Using Semantics of Natural Language*. Ph.D. thesis, Technische Universität Darmstadt, Darmstadt.
- Sascha Rothe and Hinrich Schütze. 2015. Autoextend: Extending word embeddings to embeddings for synsets and lexemes. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1793–1803, Beijing, China. Association for Computational Linguistics.
- Eugen Ruppert, Manuel Kaufmann, Martin Riedl, and Chris Biemann. 2015. Jobimviz: A web-based visualization for graph-based distributional semantic models. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 103–108, Beijing, China. Association for Computational Linguistics and The Asian Federation of Natural Language Processing.
- Hinrich Schütze. 1998. Automatic Word Sense Discrimination. *Computational Linguistics*, 24(1):97–123.
- Fei Tian, Hanjun Dai, Jiang Bian, Bin Gao, Rui Zhang, Enhong Chen, and Tie-Yan Liu. 2014. A probabilistic model for learning multi-prototype word embeddings. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING'2014)*, pages 151–160, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.
- Tim Van de Cruys. 2010. Mining for meaning: The extraction of lexicosemantic knowledge from text. *Groningen Dissertations in Linguistics*, 82.
- Stijn Van Dongen. 2008. Graph clustering via a discrete uncoupling process. *SIAM Journal on Matrix Analysis and Applications*, 30(1):121–141.

Alfredo Vellido, José David Martín, Fabrice Rossi, and Paulo J.G. Lisboa. 2011. Seeing is believing: The importance of visualization in real-world machine learning applications. In *Proceedings of the 19th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN'2011)*, pages 219–226, Bruges, Belgium.

Alfredo Vellido, José D. Martín-Guerrero, and Paulo J.G. Lisboa. 2012. Making machine learning models interpretable. In *20th European Symposium on Artificial Neural Networks, ESANN*, volume 12, pages 163–172, Bruges, Belgium.

Jean Véronis. 2004. HyperLex: Lexical cartography for information retrieval. *Computer Speech and Language*, 18:223–252.

Heng Low Wee. 2010. Word Sense Prediction Using Decision Trees. Technical report, Department of Computer Science, National University of Singapore.

Dominic Widdows and Beate Dorow. 2002. A graph model for unsupervised lexical acquisition. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING'2002)*, pages 1–7, Taipei, Taiwan. Association for Computational Linguistics.

Deniz Yuret. 2012. FASTSUBS: An efficient and exact procedure for finding the most likely lexical substitutes based on an n-gram language model. *IEEE Signal Processing Letters*, 19(11):725–728.

Zhi Zhong and Hwee Tou Ng. 2010. It makes sense: A wide-coverage word sense disambiguation system for free text. In *Proceedings of the ACL 2010 System Demonstrations*, pages 78–83, Uppsala, Sweden. Association for Computational Linguistics.

A.13 Paper “Improving Hypernymy Extraction with Distributional Semantic Classes”

A. Panchenko, D. Ustalov, S. Faralli, S. P. Ponzetto, and C. Biemann, “**Improving Hypernymy Extraction with Distributional Semantic Classes**,” in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, (Miyazaki, Japan), European Language Resources Association (ELRA), May 2018

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/L18-1244>

Improving Hypernymy Extraction with Distributional Semantic Classes

Alexander Panchenko^{1*}, Dmitry Ustalov^{2,3*}, Stefano Faralli³, Simone P. Ponzetto³, Chris Biemann¹

* these authors contributed equally

¹ University of Hamburg, Department of Informatics, Language Technology Group, Germany

² University of Mannheim, School of Business Informatics and Mathematics, Data and Web Science Group, Germany

³ Ural Federal University, Institute of Natural Sciences and Mathematics, Russia

{panchenko, biemann}@informatik.uni-hamburg.de
{dmitry, stefano, simone}@informatik.uni-mannheim.de

Abstract

In this paper, we show how distributionally-induced semantic classes can be helpful for extracting hypernyms. We present methods for inducing sense-aware semantic classes using distributional semantics and using these induced semantic classes for filtering noisy hypernymy relations. Denoising of hypernyms is performed by labeling each semantic class with its hypernyms. On the one hand, this allows us to filter out wrong extractions using the global structure of distributionally similar senses. On the other hand, we infer missing hypernyms via label propagation to cluster terms. We conduct a large-scale crowdsourcing study showing that processing of automatically extracted hypernyms using our approach improves the quality of the hypernymy extraction in terms of both precision and recall. Furthermore, we show the utility of our method in the domain taxonomy induction task, achieving the state-of-the-art results on a SemEval'16 task on taxonomy induction.

Keywords: semantic classes, distributional semantics, hypernyms, co-hyponyms, word sense induction

1. Introduction

Hypernyms are useful in various applications, such as question answering (Zhou et al., 2013), query expansion (Gong et al., 2005), and semantic role labelling (Shi and Mihalcea, 2005) as they can help to overcome sparsity of statistical models. Hypernyms are also the building blocks for learning taxonomies from text (Bordea et al., 2016). Consider the following sentence: “This café serves fresh *mangosteen* juice”. Here the infrequent word “mangosteen” may be poorly represented or even absent in the vocabulary of a statistical model, yet it can be substituted by lexical items with better representations, which carry close meaning, such as its hypernym “fruit” or one of its close co-hyponyms, e.g. “mango”.

Currently available approaches to hypernymy extraction focus on the acquisition of individual binary hypernymy relations (Hearst, 1992; Snow et al., 2004; Weeds et al., 2014; Schwartz et al., 2016; Glavaš and Ponzetto, 2017). Frequencies of the extracted relations usually follow a power-law, with a long tail of noisy extractions containing rare words. We propose a method that performs post-processing of such noisy binary hypernyms using distributional semantics, cf. Figure 1. Namely, we use the observation that distributionally related words are often co-hyponyms (Wandmacher, 2005; Heylen et al., 2008) and operationalize it to perform filtering of noisy relations by finding dense graphs composed of both hypernyms and co-hyponyms.

The contribution of the paper is an unsupervised method for post-processing of noisy hypernymy relations based on clustering of graphs of word senses induced from text. The idea to use distributional semantics to find hypernyms seems natural and has been widely used. However, the existing methods used distributional, yet *sense-unaware* and *local* features. We are the first to use *global sense-aware distributional structure* via the induced semantic classes

to improve hypernymy extraction. The implementation of our method and the induced language resources (distributional semantic classes and cleansed hypernymy relations) are available online.¹

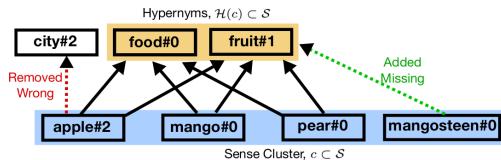


Figure 1: Our approach performs post-processing of hypernymy relations using distributionally induced semantic classes, represented by clusters of induced word senses labeled with noisy hypernyms. The word postfix, such as #1, is an ID of an induced sense. The wrong hypernyms outside the cluster labels are removed, while the missing ones not present in the noisy database of hypernyms are added.

2. Related Work

2.1. Extraction of Hypernyms

In her pioneering work, Hearst (1992) proposed to extract hypernyms based on lexical-syntactic patterns from text. Snow et al. (2004) learned such patterns automatically based on a set of hyponym-hypernym pairs. Pantel and Pennacchiotti (2006) presented another approach for weakly supervised extraction of similar extraction patterns. These approaches use some training pairs of hypernyms to bootstrap the pattern discovery process. For instance, Tjong Kim Sang (2007) used web snippets as a corpus for extraction of hypernyms. More recent approaches exploring the use of distributional word representations for extraction of

¹<https://github.com/uhh-lt/mangosteen>

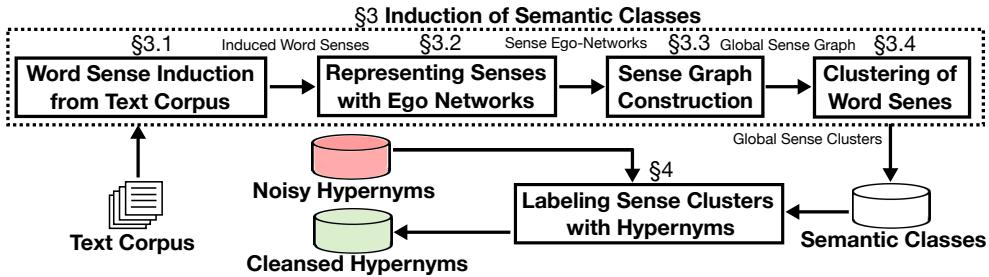


Figure 2: Outline of our approach: sense-aware distributional semantic classes are induced from a text corpus and then used to filter noisy hypernyms database (e.g. extracted by an external method from a text corpus).

hypernyms and co-hypernyms include (Roller et al., 2014; Weeds et al., 2014; Neculescu et al., 2015; Vylomova et al., 2016). They rely on two distributional vectors to characterize a relation between two words, e.g. on the basis of the difference of such vectors or their concatenation. Levy et al. (2015) discovered a tendency to lexical memorization of such approaches, hampering their generalization to other domains.

Fu et al. (2014) relied on an alternative approach where a projection matrix is learned, which transforms a distributional vector of a hyponym to the vector of its hypernym. Ustalov et al. (2017a) improved this method by adding regularizers in the model that take into account negative training samples and the asymmetric nature of the hypernyms. Recent approaches to hypernym extraction focused on learning *supervised* models based on a combination of syntactic patterns and distributional features (Shwartz et al., 2016). Note that while methods, such as (Mirkin et al., 2006) and (Shwartz et al., 2016) use distributional features for extraction of hypernyms, in contrast to our method, they do not take into account word senses and global distributional structure.

Seitner et al. (2016) performed extraction of hypernyms from the web-scale Common Crawl² text corpus to ensure high lexical coverage. In our experiments, we use this web-scale database of noisy hypernyms, as the large-scale repository of automatically extracted hypernyms to date.

2.2. Taxonomy and Ontology Learning

Most relevant in the context of automatic construction of lexical resource are methods for building resources from text (Caraballo, 1999; Biemann, 2005; Cimiano, 2006; Bordea et al., 2015; Velardi et al., 2013) as opposed to methods that automatically construct resources from semi-structured data (Auer et al., 2007; Navigli and Ponzetto, 2012) or using crowdsourcing (Biemann, 2013; Braslavski et al., 2016).

Our representation differs from the global hierarchy of words as constructed e.g. by (Berant et al., 2011; Faralli et al., 2016), as we are grouping many lexical items into a labeled sense cluster as opposed to organizing them in deep hierarchies. Kozareva and Hovy (2013) proposed a taxonomy induction method based on extraction of hypernyms using the doubly-anchored lexical patterns. Graph

algorithms are used to induce a proper tree from the binary relations harvested from text.

2.3. Induction of Semantic Classes

This line of research starts with (Lin and Pantel, 2001), where sets of similar words are clustered into concepts. While this approach performs a hard clustering and does not label clusters, these drawbacks are addressed in (Pantel and Lin, 2002), where words can belong to several clusters, thus representing senses, and in (Pantel and Ravichandran, 2004), where authors aggregate hypernyms per cluster, which come from Hearst patterns. The main difference to our approach is that we explicitly represent senses both in clusters and in their hypernym labels, which enables us to connect our sense clusters into a global taxonomic structure. Consequently, we are the first to use semantic classes to improve hypernym extraction.

Ustalov et al. (2017b) proposed a synset induction approach based on global clustering of word senses. The authors used the graph constructed of dictionary synonyms, while we use distributionally-induced graphs of senses.

3. Unsupervised Induction of Distributional Sense-Aware Semantic Classes

As illustrated in Figure 2, our method induces a sense inventory from a text corpus using the method of (Faralli et al., 2016; Biemann et al., 2018), and clusters these senses. Sample word senses from the induced sense inventory are presented in Table 1. The difference of the induced sense inventory from the sense clustering presented in Table 2 is that word senses in the induced resource are specific to a given target word, e.g. words “apple” and “mango” have distinct “fruit” senses, represented by a list of related senses. On the other hand, sense clusters represent a global and not a local clustering of senses, i.e. the “apple” in the “fruit” sense can be a member of only one cluster. This is similar to WordNet, where one sense can only belong to a single synset. Below we describe each step of our method.

3.1. Word Sense Induction from a Text Corpus

Each word sense s in the induced sense inventory \mathcal{S} is represented by a list of neighbors $\mathcal{N}(s)$, see Table 1 for an example. Extraction of this network is performed using the method of Faralli et al. (2016) and involves three steps: (1) building a distributional thesaurus, i.e. a graph

²<http://www.commoncrawl.org>

ID: Word Sense, $s \in \mathcal{S}$	Local Sense Cluster: Related Senses, $\mathcal{N}(s) \subset \mathcal{S}$	Hypernyms, $\mathcal{H}(s) \subset \mathcal{S}$
mango#0	peach#1, grape#0, plum#0, apple#0, apricot#0, watermelon#1, banana#1, coconut#0, pear#0, fig#0, melon#0, mangosteen#0 , ...	fruit#0, food#0, ...
apple#0	mango#0, pineapple#0, banana#1, melon#0, grape#0, peach#1, watermelon#1, apricot#0, cranberry#0, pumpkin#0, mangosteen#0 , ...	fruit#0, crop#0, ...
Java#1	C#4, Python#3, Apache#3, Ruby#6, Flash#1, C++#0, SQL#0, ASP#2, Visual Basic#1, CSS#0, Delphi#2, MySQL#0, Excel#0, Pascal#0, ...	programming language#3, language#0, ...
Python#3	PHP#0, Pascal#0, Java#1, SQL#0, Visual Basic#1, C++#0, JavaScript#0, Apache#3, Haskell#5, .NET#1, C#4, SQL Server#0, ...	language#0, technology#0, ...

Table 1: Sample induced sense inventory entries representing “fruits” and “programming language” senses. Each word sense s is represented with a list of related senses $\mathcal{N}(s)$ and the list of hypernyms $\mathcal{H}(s)$. The hypernyms can be used as human-interpretable sense labels of the sense clusters. One sense s , such as “apple#0”, can appear in multiple entries.

ID	Global Sense Cluster: Semantic Class, $c \subset \mathcal{S}$	Hypernyms, $\mathcal{H}(c) \subset \mathcal{S}$
1	peach#1, banana#1, pineapple#0, berry#0, blackberry#0, grapefruit#0, strawberry#0, blueberry#0, fruit#0, grape#0, melon#0, orange#0, pear#0, plum#0, raspberry#0, watermelon#0, apple#0, apricot#0, watermelon#0, pumpkin#0, berry#0, mangosteen#0 , ...	vegetable#0, fruit#0, crop#0, ingredient#0, food#0, ...
2	C#4, Basic#2, Haskell#5, Flash#1, Java#1, Pascal#0, Ruby#6, PHP#0, Ada#1, Oracle#3, Python#3, Apache#3, Visual Basic#1, ASP#2, Delphi#2, SQL Server#0, CSS#0, AJAX#0, JavaScript#0, SQL Server#0, Apache#3, Delphi#2, Haskell#5, .NET#1, CSS#0, ...	programming language#3, technology#0, language#0, format#2, app#0

Table 2: Sample of the induced sense clusters representing “fruits” and “programming language” semantic classes. Similarly to the induced word senses, the semantic classes are labeled with hypernyms. In contrast to the induced word senses, which represent a local clustering of word senses (related to a given word) semantic classes represent a global sense clustering of word senses. One sense c , such as “apple#0”, can appear only in a single cluster.

of related ambiguous terms (Biemann and Riedl, 2013); (2) word sense induction via clustering of ego networks (Widows and Dorow, 2002; Everett and Borgatti, 2005) of related words using the Chinese Whispers graph clustering algorithm (Biemann, 2006); (3) disambiguation of related words and hypernyms. The word sense inventory used in our experiment³ was extracted from a 9.3 billion tokens corpus, which is a concatenation of Wikipedia⁴, ukWac (Ferraresi et al., 2008), LCC (Richter et al., 2006) and Gigaword (Graff and Cieri, 2003). Note that analogous graphs of senses can be obtained using word sense embeddings, see (Neelakantan et al., 2014; Bartunov et al., 2016). Similarly to any other distributional word graph, the induced sense inventory sense network is scale-free, cf. (Steyvers and Tenenbaum, 2005). Our experiments show that a global clustering of this network can lead to a discovery of giant components, which are useless in our context as they represent no semantic class. To overcome this problem, we re-build the sense network as described below.

3.2. Representing Senses with Ego Networks

To perform a global clustering of senses, we represent each induced sense s by a second-order *ego network* (Everett and Borgatti, 2005). An ego network is a graph consisting of all related senses $\mathcal{R}(s)$ of the ego sense s reachable via a path of length one or two, defined as:

$$\{s_j : (s_j \in \mathcal{N}(s)) \vee (s_i \in \mathcal{N}(s) \wedge s_j \in \mathcal{N}(s_i))\}. \quad (1)$$

Each edge weight $\mathcal{W}_s(s_i, s_j)$ between two senses is taken

from the induced sense inventory network (Faralli et al., 2016) and is equal to a distributional semantic relatedness score between s_i and s_j .

Senses in the induced sense inventory may contain a mixture of different senses introducing noise in a global clustering: cf. Figure 3, where “Python” in the animal sense is related to both car and snake senses. To minimize the impact of the word sense induction errors, we filter out ego networks with a highly segmented structure. Namely, we cluster each ego network with the Chinese Whispers algorithm and discard networks for which the cluster containing the target sense s contains less than 80% nodes of the respective network to ensure semantic coherence inside the word groups. Besides, all nodes of a network not appearing in the cluster containing the ego sense s are also discarded.

3.3. Global Sense Graph Construction

The goal of this step is to merge ego networks of individual senses constructed at the previous step into a global graph. We compute weights of the edges of the global graph by counting the number of co-occurrences of the same edge in different networks:

$$\mathcal{W}(s_i, s_j) = \sum_{s \in \mathcal{S}} \mathcal{W}_s(s_i, s_j). \quad (2)$$

For filtering out noisy edges, we remove all edges with the weight less than a threshold t . Finally, we apply the function $E(w)$ that re-scales edge weights. We tested identity function (count) and the natural logarithm (log):

$$\mathcal{W}(s_i, s_j) = \begin{cases} E(\mathcal{W}(s_i, s_j)) & \text{if } \mathcal{W}(s_i, s_j) \geq T, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

³The input and output datasets are available for download at <https://doi.org/10.5281/zenodo.1174041>

⁴<https://doi.org/10.5281/zenodo.229904>

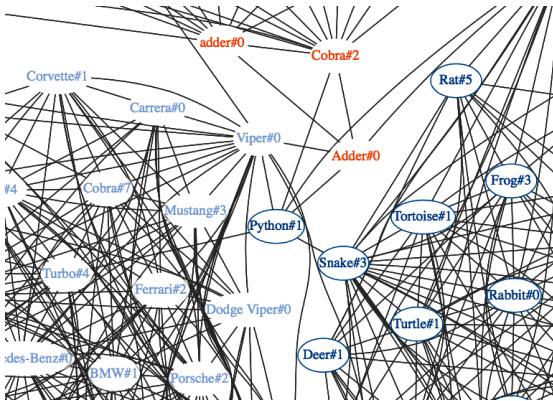


Figure 3: An example of a non-coherent ego network of the automatically induced sense Python#1, representing the “animal” sense. We prune it to remove terms not relevant to the animal sense.

3.4. Clustering of Word Senses

The core of our method is the induction of semantic classes by clustering the global graph of word senses. We use the Chinese Whispers algorithm to make every sense appear only in one cluster c . Results of the algorithm are groups of strongly related word senses that represent different concepts (cf. Figure 4). Hypernymy is by definition a relation between nouns. Thus optionally, we remove all single-word senses that do not correspond to nouns using the Pattern library (De Smedt and Daelemans, 2012). This optional mode is configured by the boolean parameter N .

We use two clustering versions in our experiments: the *fine-grained* model clusters 208,871 induced word senses into 1,870 semantic classes, and the *coarse-grained* model that groups 18,028 word senses into 734 semantic classes. To find optimal parameters of our method, we compare the induced labeled sense clusters to lexical semantic knowledge from WordNet 3.1 (Fellbaum, 1998) and BabelNet 3.7 (Navigli and Ponzetto, 2012).

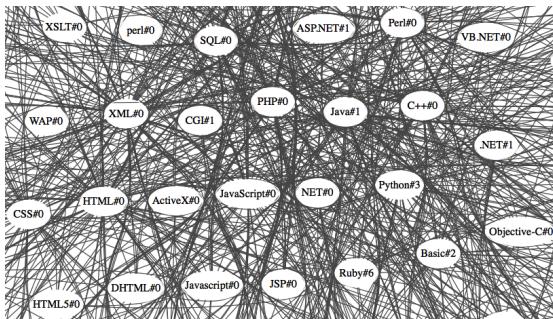


Figure 4: Senses referring to programming languages co-occur in global sense cluster entries, resulting in a densely connected set of co-hyponyms.

4. Denoising Hypernyms using the Induced Distributional Semantic Classes

By labeling the induced semantic classes with hypernyms we can thereby remove wrong ones or add those that are missing as illustrated in Figure 1. Each sense cluster is labeled with the noisy input hypernyms, where the labels are the common hypernyms of the cluster word (cf. Table 2). Hypernyms that label no sense cluster are filtered out. In addition, new hypernyms can be generated as a result of labeling. Additional hypernyms are discovered by propagating cluster labels to the rare words without hypernyms, e.g. “mangosteen” in Figure 1. For labeling we used the tf-idf weighting. Hypernyms that appear in many senses s are weighted down:

$$\text{tf-idf}(h) = \sum_{s \in c} \mathcal{H}(s) \cdot \log \frac{|\mathcal{S}|}{|h \in \mathcal{H}(s) : \forall s \in \mathcal{S}|}, \quad (4)$$

where $\sum_{s \in c} \mathcal{H}(s)$ is a sum of weights for all hypernyms for each sense s , per each cluster c .

We label each sense cluster c with its top five hypernyms $\mathcal{H}(c)$. Each hypernym is disambiguated using the method of Faralli et al. (2016). Namely, we calculate the cosine similarity between the context (the current sense cluster) and the induced senses (local clusters of the ambiguous word).

Distributional representations of rare words, such as “mangosteen” can be less precise than those of frequent words. However, co-occurrence of a hyponym and a hypernym in a single sentence is not required in our approach, while it is the case for the path-based hypernymy extraction methods.

5. Finding an Optimal Configuration of Meta Parameters of the Method

The approach consists of several sequential stages, as depicted in Figure 2, with each stage having a few meta parameters. This study is designed to find promising combinations of these meta parameters. In this section, we propose several metrics which aim at finding an optimal configuration of all these meta parameters jointly. In particular, to compare different configurations of our approach, we compare the labeled sense clusters to WordNet 3.1 (Fellbaum, 1998) and BabelNet 3.7 (Navigli and Ponzetto, 2012). The assumption is that the optimal model contains lexical semantic knowledge similar to the knowledge in the lexical resources. To implement the evaluation metrics we used the NLTK library (Bird et al., 2009) and the BabelNet Java API.⁵

5.1. Metrics Quantifying Goodness of Fit of the Induced Structures to Lexical Resources

To summarize various aspects of the lexical resource, we propose a score that is maximized if labeled sense clusters are generated directly from a lexical resource:

$$hpc\text{-score}(c) = \frac{h\text{-score}(c) + 1}{p\text{-score}(c) + 1} \cdot \text{coverage}(c). \quad (5)$$

$p\text{-score}(c)$ quantifies the plausibility of the sense cluster c .

⁵<http://www.babelnet.org>

	Min. sense co-occurrences, t	Edge weight, E	Only nouns, N	Hypernym weight, H	Number of clusters	Number of senses	$hpc\text{-avg}$, WordNet	$hpc\text{-avg}$, BabelNet
coarse-gr.	100	log	yes	tf-idf	734	18 028	0.092	0.304
	100	log	no	tf-idf	763	27 149	0.090	0.303
	100	count	no	tf-idf	765	27 149	0.089	0.302
	100	log	no	tf	784	27 149	0.090	0.300
	100	count	yes	tf	733	18 028	0.092	0.299
	100	count	no	tf	772	27 149	0.089	0.297
	100	count	yes	tf-idf	732	18 028	0.091	0.295
	100	log	yes	tf	726	18 028	0.088	0.293
fine-gr.	0	count	no	tf-idf	1870	208 871	0.041	0.279
	0	count	no	tf	1877	208 871	0.041	0.278
	0	count	yes	tf	2070	144 336	0.037	0.240
	0	count	yes	tf-idf	2080	144 336	0.038	0.240
	0	log	yes	tf-idf	4709	144 336	0.027	0.138
	0	log	yes	tf	4679	144 336	0.027	0.136
	0	log	no	tf-idf	5960	208 871	0.035	0.127
	0	log	no	tf	5905	208 871	0.036	0.126

Table 3: Performance of different configurations of the hypernymy labeled global sense clusters in terms of their similarity to WordNet/BabelNet. The results are sorted by performance on BabelNet dataset, the best values in each section are boldfaced. The two underlined configurations are respectively the best *coarse-grained* and *fine-grained* grained semantic class models used in all experiments. The coarse grained model contains less semantic classes, but they tend to be more consistent than those of the fine-grained model, which contains more senses and classes.

It reflects the distance of co-hyponyms in a lexical resource:

$$p\text{-score}(c) = \frac{1}{|c|} \sum_{i=1}^{|c|} \sum_{j=1}^i \text{dist}(w_i, w_j). \quad (6)$$

The lower the p -score is, the closer the hyponyms are located in the gold standard resource. For each pair of distinct lemmas (w_i, w_j) in the cluster of co-hyponyms c , we search for the minimal shortest path distance (SPD) between the synsets corresponding to each word in the pair, i.e. $S(w_i)$ is the set of synsets having the w_i lemma and $S(w_j)$ is the similar set with respect to the w_j lemma:

$$\text{dist}(w_i, w_j) = \min_{\substack{s' \in S(w_i), \\ s'' \in S(w_j)}} \text{SPD}(s', s''). \quad (7)$$

$h\text{-score}(c)$ quantifies plausibility of the hypernyms $\mathcal{H}(c)$ of a sense cluster c measuring the precision of extracted hypernyms:

$$h\text{-score}(c) = \frac{|\mathcal{H}(c) \cap \text{gold}(c)|}{|\mathcal{H}(c)|}. \quad (8)$$

The $\text{gold}(c)$ is composed of the lowest common hypernyms (LCH) in the lexical resource for each pair of lemmas in the sense cluster c :

$$\text{gold}(c) = \bigcup_{w_i \in c} \bigcup_{\substack{s' \in S(w_i), \\ w_j \in c \\ s'' \in S(w_j)}} \{\text{LCH}(s', s'')\}. \quad (9)$$

$\text{coverage}(c)$ quantifies how well cluster words are represented in the gold standard resource. Thus, errors in poorly represented clusters are discounted via coverage. Coverage is the fraction of the lemmas appearing both in the cluster c and in the vocabulary of the resource \mathcal{V} :

$$\text{coverage}(c) = \frac{|c \cap \mathcal{V}|}{|c|}. \quad (10)$$

The total score used to rank various configurations of our approach averages hpc -score scores for all induced sense

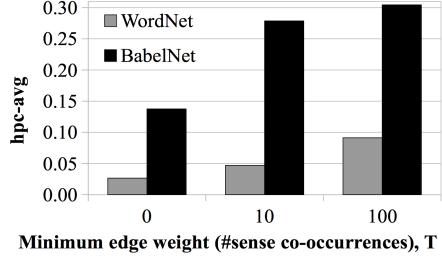


Figure 5: Impact of the min. edge weight t .

clusters:

$$hpc\text{-avg} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} hpc\text{-score}(c). \quad (11)$$

5.2. Results

Meta parameter search results based on the comparison to WordNet and BabelNet are provided in Figure 5 and Table 3. The minimal edge weight t trades off between the size of the resulting resource (number of words and senses) and its similarity to the gold lexical resources. The higher the threshold, the fewer nodes remain in the graph, yet these remaining nodes form densely interlinked communities. For t of 100, each pair of senses in the graph is observed in at least 100 ego networks. Secondly, for the unpruned model ($t = 0$), edge weights based on counts worked better than logarithmic weights. However, when pruned ($t > 0$), logarithmic edge weighting shows better results. Thirdly, the tf-idf weights proved to yield consistent improvements over the basic tf weighting. For the pruned model, the variation in scores across different configurations is small as the underlying graphs are of high quality, while for the unpruned model the choice of parameters has

much more impact as the sense graphs are noisier. We selected the best-performed configuration according to BabelNet (*hpc*-avg of 0.304), which also is the second best configuration according to WordNet (*hpc*-avg of 0.092). This model is based on the edge threshold t of 100, logarithmic weights of edges contains only nouns and hypernyms ranked according to tf-idf. Note also that, the best-unpruned model ($t = 0$) has BabelNet *hpc*-avg of 0.279, which is only 10% lower than the best model, yet the unpruned model has an order of magnitude larger vocabulary and a more fine-grained representation (734 vs. 1,870 clusters). Thus, if coverage is important, the unpruned model is recommended. In the remainder of this paper, we continue with the first model listed in Table 3 and evaluate it in the following experiments.

6. Evaluation

To evaluate our approach, we conduct two intrinsic evaluations and one extrinsic evaluation. The first experiment aims to estimate the fraction of spurious sense clusters, the second one evaluates the quality of the post-processed hypernyms. Finally, we evaluate the induced semantic classes in application to the taxonomy induction task.

6.1. Experiment 1: Plausibility of the Induced Semantic Classes

Comparison to gold standard resources allows us to gauge the relative performances of various configurations of our method. To measure the absolute quality of the best configuration selected in the previous section, we rely on microtask-based crowdsourcing with CrowdFlower⁶.

6.1.1. Task Design

We used two crowdsourcing tasks based on word intruder detection (Chang et al., 2009) to measure how humans perceive the extracted lexical-semantic structures. Namely, the tasks are designed to evaluate the quality of the extracted sense clusters and their labels. The input form presented to an annotator is illustrated in Figure 6. A crowdworker is asked to identify words that do not match the context represented by words from a sense cluster or its label. To generate an intruder, following the original design of Chang et al. (2009), we select a random word from a cluster and replace it with a word of similar frequency that does not belong to any cluster (bias here is low as the evaluated model contains 27,149 out of 313,841 induced word senses). In both tasks, the workers have been provided with concise instructions and test questions.

6.1.2. Evaluation Metrics

We compute two metrics on the basis on annotation results: (1) *accuracy* is the fraction of tasks where annotators correctly identified the intruder, thus the words from the cluster are consistent; (2) *badness* is the fraction of tasks for which non-intruder words were selected. In this experiment, we assume that it is easy to identify the intruder in a correct sense cluster and difficult in a noisy, implausible sense cluster. We compute *accuracy* as the fraction of tasks

Topics:
• vegetable
• fruit
• crop
For these topics we have the list of the following words:
• peach
• pineapple
• winchester
• watermelon
• cherry
• blackberry
Select the words that are non-relevant for the topics above:
<input type="checkbox"/> peach
<input type="checkbox"/> pineapple
<input type="checkbox"/> winchester
<input type="checkbox"/> watermelon
<input type="checkbox"/> cherry
<input type="checkbox"/> blackberry

Figure 6: Layout of the sense cluster evaluation crowdsourcing task, the entry “winchester” is the intruder.

	Accuracy	Badness	Randolph κ
Sense clusters, c	0.859	0.248	0.739
Hyper. labels, $\mathcal{H}(c)$	0.919	0.208	0.705

Table 4: Plausibility of the sense clusters according to human judgments via an intruder detection experiment for the coarse-grained semantic class model.

where annotators correctly identified the intruder, thus the words from the cluster are consistent.

6.1.3. Results

Table 4 summarizes the results of the intruder detection experiment. Overall, 68 annotators provided 2,035 judgments about the quality of sense clusters. Regarding hypernyms, 98 annotators provided 2,245 judgments. The majority of the induced semantic classes and their labels are highly plausible according to human judgments: the accuracy of the sense clusters based on the intruder detection is 0.859 (agreement of 87%), while the accuracy of hypernyms is 0.919 (agreement of 85%). The Randolph κ of respectively 0.739 and 0.705 indicates substantial inter-observer agreement (Randolph, 2005).

According to the feedback mechanism of the CrowdFlower, the co-hyponymy task received a 4.0 out of 5.0 rating, while the hypernymy task received a 4.4 out of 5.0 rating. The crowdworkers show a *substantial* agreement according to Randolph κ coefficient computed 0.739 for the cluster evaluation task and 0.705 for the hypernym evaluation task.

Major sources of errors for crowdworkers are rare words and entities. While clusters with well-known entities, such as “Richard Nixon” and “Windows Vista” are correctly labeled, examples of other less-known named entities, e.g. cricket players, are sometimes wrongly labeled as implausible. Another source of errors during crowdsourcing were wrongly assigned hypernyms: in rare cases, sense clusters are labeled with hypernyms like “thing” or “object” that are

⁶<https://www.crowdflower.com>

	Precision	Recall	F-score
Original hypernymy relations extracted from the Common Crawl corpus (Seitner et al., 2016)	0.475	0.546	0.508
Enhanced hypernyms with the <i>coarse-grained</i> semantic classes	0.541	0.679	0.602

Table 5: Results of post-processing of a noisy hypernymy database with our approach, evaluated using human judgements.

Is it correct that **peach** is a kind of **fruit**?
Your opinion:
 Yes
 No

Figure 7: Layout of the hypernymy annotation task.

too generic even under tf-idf weighting.

6.2. Experiment 2: Improving Binary Hypernymy Relations

In this experiment, we test whether our post-processing based on the semantic class improves the quality of hypernymy relations (cf. Figure 2).

6.2.1. Generation of Binary Hypernyms.

We evaluated the best coarse-grained model identified in the first experiment (t of 100). Each sense cluster of this model is split into the set $H_{cluster}$ of binary hypernyms, as illustrated in Figure 1. Overall, we gathered 85,290 hypernym relations for 17,058 unique hyponyms. Next, we gathered the set H_{orig} of 75,486 original hypernyms for exactly the same 17,058 hyponyms. For each word from the sense cluster we looked up top five hypernyms under the best ones when sorting them by extraction frequency from the hypernym relation database of Seitner et al. (2016) as in our model each sense cluster is labeled with five hypernyms from the same database. The database of Seitner et al. (2016) is extracted using lexical patterns. Note that any other method for extraction of binary hypernyms can be used at this point, e.g. (Weeds et al., 2014; Roller et al., 2014; Shwartz et al., 2016; Glavaš and Ponzetto, 2017). For the comparison, we gathered up to five hypernyms for each word, using (1) the most frequent hypernym relations from (Seitner et al., 2016) vs. (2) the cluster labeling method as described above.

6.2.2. Task Design

We drew a random sample of 4,870 relations using lexical split by hyponyms. All relations from $H_{cluster}$ and H_{orig} of one hyponym were included in the sample. These relations were subsequently annotated by human judges using crowdsourcing. We asked crowdworkers to provide a binary judgment about the correctness of each hypernymy relation as illustrated in Figure 7.

6.2.3. Results

Overall, 298 annotators completed 4,870 unique tasks each labeled 6.9 times on average, resulting in a total of 33,719 binary human judgments about hypernyms. We obtained a *fair* agreement among annotators of 0.548 in terms of the Randolph κ (Meyer et al., 2014). Since CrowdFlower reports a confidence for each answer, we selected $N = 3$

most confident answers per pair and aggregated them using weighted majority voting. The ties were broken pessimistically, i.e. by treating a hypernym as irrelevant. Results for $N \in 3, 5, 6$ varied less than by 0.002 in terms of F-score. The task received the rating of a 4.4 out of 5.0 according to the annotator’s feedback mechanism.

Table 5 presents results of the experiment. Since each pair received a binary score, we calculated Precision, Recall, and F-measure of two compared methods. Our denoising method improves the quality of the original hypernyms by a large margin both in terms of precision and recall, leading to an overall improvement of 10 F-score points. The improvements of recall are due to the fact that to label a cluster of co-hyponyms it is sufficient to lookup hypernyms for only a fraction of words in the clusters. However, binary relations will be generated between all cluster hypernyms and the cluster words potentially generating hypernyms missing in the input database. For instance, a cluster of fruits can contain common entries like “apple” and “mango” which ensure labeling it with the word “fruit”. Rare words in the same cluster, like “mangosteen”, which have no hypernyms in the original resource due to the sparsity of the pattern-based approach, will also obtain the hypernym “fruit” as they are distributionally related to frequent words with reliable hypernym relations, cf. Figure 1. We also observed this effect frequently with clusters of named entities, like cricket players. Improvements in precision are due to filtering of wrong extractions, which are different for different words and thus top hypernyms of a cluster contain only hypernyms confirmed by several co-hyponyms.

Finally, note that all previous hypernymy extraction methods output binary relations between undisambiguated words (cf. Section 2.). Therefore, our approach could be used to improve results of other state-of-the-art hypernymy extraction approaches, such as HypeNET (Shwartz et al., 2016).

6.3. Experiment 3: Improving Domain Taxonomy Induction

In this section, we show how the labeled semantic classes can be used for induction of domain taxonomies.

6.3.1. SemEval 2016 Task 13

We use the taxonomy extraction evaluation dataset by Bordea et al. (2016), featuring gold standard taxonomies for three domains (Food, Science, Environment) and four languages (English, Dutch, French, and Italian) on the basis of existing lexical resources, such as WordNet and Eurovoc (Steinberger et al., 2006).⁷ Participants were supposed to build a taxonomy provided a vocabulary of a domain. Since our other experiments were conducted on English, we used the English part of the task. The evaluation is

⁷<http://eurovoc.europa.eu>

System / Domain, Dataset	Food, WordNet	Science, WordNet	Food, Combined	Science, Combined	Science, Eurovoc	Environment, Eurovoc
WordNet	1.0000	1.0000	0.5870	0.5760	0.6243	n.a.
Baseline	0.0022	0.0016	0.0019	0.0163	0.0056	0.0000
JUNLP	0.1925	0.0494	0.2608	0.1774	0.1373	0.0814
NUIG-UNLP	n.a.	0.0027	n.a.	0.0090	0.1517	0.0007
QASSIT	n.a.	0.2255	n.a.	0.5757	0.3893	0.4349
TAXI	0.3260	0.2255	0.2021	0.3634	0.3893	0.2384
USAAR	0.0021	0.0008	0.0000	0.0020	0.0023	0.0007
Semantic Classes (fine-grained)	0.4540	0.4181	0.5147	0.6359	0.5831	0.5600
Semantic Classes (coarse-grained)	0.4774	0.5927	0.5799	0.6539	0.5515	0.6326

Table 6: Comparison of our taxonomy induction method on the SemEval 2016 Task 13 on Taxonomy Extraction Evaluation (Bordea et al., 2016) for English in terms of cumulative Fowlkes&Mallows measure (F&M).

Domain	#Seeds words	#Expand. words	#Clusters, fine-gr.	#Clusters, coarse-gr.
Food	2 834	3 047	29	21
Science	806	1 137	73	35
Environ.	261	909	111	39

Table 7: Summary of the domain-specific sense clusters.

based on the Fowlkes&Mallows Measure (F&M), a cumulative measure of the similarity of both taxonomies (Velardi et al., 2013).

6.3.2. Taxonomy Induction using Semantic Classes

Our method for taxonomy induction takes as input a vocabulary of the domain and outputs a taxonomy of the domain. The method consists of three steps: (1) retrieving sense clusters relevant to the target domain; (2) generation of binary relations through a Cartesian product of words in a sense cluster and its labels; (3) attaching disconnected components to the root (the name of the domain). We retrieve domain-specific senses for each domain of the SemEval datasets by a lexical filtering. First, we build an extended lexicon of each domain on the basis of the seed vocabulary of the domain provided in the SemEval dataset. Namely, for each seed term, we retrieve all semantically similar terms. To filter out noisy expansions, related terms are added to the expanded vocabulary only if there are at least $k = 5$ common terms between the seed vocabulary and the list of related terms. Second, we retrieve all sense clusters that contain at least one term from the expanded vocabulary among its sense clusters or hypernyms. Table 7 summarizes results of this domain filtering. After, we generate binary hypernymy relations by linking every word in the semantic class to each hypernymy label as shown in Figure 1. Finally, we link roots of each disconnected components to the root of the taxonomy, e.g. “food” for the Food domain. Note that this step was used by SemEval participants, e.g. in the TAXI system (Panchenko et al., 2016).

6.3.3. Results

Table 6 presents results of the taxonomy extraction experiment. We evaluated two best models of our method: a *coarse* and a *fine grained* clusterings featuring respectively 734 and 1870 semantic classes identified in Section 5. with different levels of pruning: $t \in \{0, 100\}$. As one

can observe, our model based on the labeled sense clusters significantly outperforms the substring-based baseline and all participating system by a large margin on all domains. For the “Science (Eurovoc)” and “Food” domains our method yields results comparable to WordNet while remaining unsupervised and knowledge-free. Besides, for the “Science” domain our method outperforms WordNet, indicating on the high quality of the extracted lexical semantic knowledge. Overall, the *coarse-grained* more pruned model yielded better results as compared to *fine-grained* un-pruned model for all domains but “Science (Eurovoc)”.

7. Conclusion

In this paper, we presented an unsupervised method for the induction of sense-aware semantic classes using distributional semantics and graph clustering and showed how these can be used for post-processing of noisy hypernymy databases extracted from text. We determined optimal parameters of our approach by a comparison to existing lexical-semantic networks. To evaluate our approach, we performed three experiments. A large-scale crowdsourcing study indicated a high plausibility of extracted semantic classes according to human judgment. Besides, we demonstrated that our approach helps to improve precision and recall of a hypernymy extraction method. Finally, we showed how the proposed semantic classes can be used to improve domain taxonomy induction from text.

While we have demonstrated the utility of our approach for hypernym extraction and taxonomy induction, we believe that the induced semantic classes can be useful in other tasks. For instance, in (Panchenko et al., 2017) these semantic classes were used as an inventory for word sense disambiguation to deal with out-of-vocabulary words.

8. Acknowledgements

This research was supported by Deutscher Akademischer Austauschdienst (DAAD), Deutsche Forschungsgemeinschaft (DFG) under the project “Joining Ontologies and Semantics Induced from Text” (JOIN-T), and by the Ministry of Education and Science of the Russian Federation Agreement no. 02.A03.21.0006. We are grateful to three anonymous reviewers for their helpful comments. Finally, we are grateful to Dirk Johannßen for providing feedback on an early version of this paper.

9. Bibliographical References

- Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). DBpedia: A Nucleus for a Web of Open Data. In *Proceedings of the 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007: The Semantic Web*, pages 722–735. Springer Berlin Heidelberg, Busan, Korea.
- Bartunov, S., Kondrashkin, D., Osokin, A., and Vetrov, D. (2016). Breaking sticks and ambiguities with adaptive skip-gram. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 130–138, Cadiz, Spain.
- Berant, J., Dagan, I., and Goldberger, J. (2011). Global Learning of Typed Entailment Rules. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 610–619, Portland, Oregon, USA. Association for Computational Linguistics.
- Biemann, C. and Riedl, M. (2013). Text: now in 2D! A framework for lexical expansion with contextual similarity. *Journal of Language Modelling*, 1(1):55–95.
- Biemann, C., Faralli, S., Panchenko, A., and Ponzetto, S. P. (2018). A framework for enriching lexical semantic resources with distributional semantics. *Natural Language Engineering*, pages 1–48.
- Biemann, C. (2005). Ontology Learning from Text: A Survey of Methods. *GLDV-Journal for Computational Linguistics and Language Technology*, 20(2):75–93.
- Biemann, C. (2006). Chinese Whispers: An Efficient Graph Clustering Algorithm and Its Application to Natural Language Processing Problems. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing, TextGraphs-1*, pages 73–80, New York, NY, USA. Association for Computational Linguistics.
- Biemann, C. (2013). Creating a system for lexical substitutions from scratch using crowdsourcing. *Language Resources and Evaluation*, 47(1):97–122.
- Bird, S., Loper, E., and Klein, E. (2009). *Natural Language Processing with Python*. O'Reilly Media Inc.
- Bordea, G., Buitelaar, P., Faralli, S., and Navigli, R. (2015). SemEval-2015 Task 17: Taxonomy Extraction Evaluation (TExEval). In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 902–910, Denver, CO, USA, June. Association for Computational Linguistics.
- Bordea, G., Lefever, E., and Buitelaar, P. (2016). SemEval-2016 task 13: Taxonomy extraction evaluation (texeval-2). In *SemEval-2016*, pages 1081–1091, San Diego, CA, USA. Association for Computational Linguistics.
- Braslavski, P., Ustalov, D., Mukhin, M., and Kiselev, Y. (2016). YARN: Spinning-in-Progress. In *Proceedings of the 8th Global WordNet Conference, GWC 2016*, pages 58–65, Bucharest, Romania. Global WordNet Association.
- Caraballo, S. A. (1999). Automatic construction of a hypernym-labeled noun hierarchy from text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 120–126, College Park, MD, USA. Association for Computational Linguistics.
- Chang, J., Boyd-Graber, J., Gerrish, S., Wang, C., and Blei, D. M. (2009). Reading Tea Leaves: How Humans Interpret Topic Models. In *Advances in Neural Information Processing Systems 22*, pages 288–296. Curran Associates, Inc.
- Cimiano, P. (2006). *Ontology Learning from Text*, pages 19–34. Springer US, Boston, MA, USA.
- De Smedt, T. and Daelemans, W. (2012). Pattern for Python. *Journal of Machine Learning Research*, 13:2031–2035.
- Everett, M. and Borgatti, S. P. (2005). Ego network betweenness. *Social networks*, 27(1):31–38.
- Faralli, S., Panchenko, A., Biemann, C., and Ponzetto, S. P. (2016). Linked Disambiguated Distributional Semantic Networks. In *The Semantic Web – ISWC 2016: 15th International Semantic Web Conference, Part II*, Lecture Notes in Computer Science, pages 56–64, Kobe, Japan. Springer International Publishing.
- Fellbaum, C. (1998). *WordNet: An Electronic Database*. MIT Press.
- Ferraresi, A., Zanchetta, E., Baroni, M., and Bernardini, S. (2008). Introducing and evaluating ukWaC, a very large web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4): Can we beat Google?*, pages 47–54, Marrakech, Morocco. European Language Resources Association (ELRA).
- Fu, R., Guo, J., Qin, B., Che, W., Wang, H., and Liu, T. (2014). Learning Semantic Hierarchies via Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1199–1209, Baltimore, MD, USA. Association for Computational Linguistics.
- Glavaš, G. and Ponzetto, S. P. (2017). Dual tensor model for detecting asymmetric lexico-semantic relations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1758–1768, Copenhagen, Denmark, September. Association for Computational Linguistics.
- Gong, Z., Cheang, C. W., and Leong Hou, U. (2005). Web Query Expansion by WordNet. In *Proceedings of the 16th International Conference on Database and Expert Systems Applications - DEXA '05*, pages 166–175, Copenhagen, Denmark. Springer Berlin Heidelberg.
- Graff, D. and Cieri, C. (2003). English Gigaword corpus. *Linguistic Data Consortium*.
- Hearst, M. A. (1992). Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of the 14th Conference on Computational Linguistics - Volume 2*, COLING '92, pages 539–545, Nantes, France. Association for Computational Linguistics.
- Heylen, K., Peirsman, Y., Geeraerts, D., and Speelman, D. (2008). Modelling Word Similarity: an Evaluation of Automatic Synonymy Extraction Algorithms. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation, LREC 2008*, pages 1–6.

- 3243–3249, Marrakech, Morocco. European Language Resources Association (ELRA).
- Kozareva, Z. and Hovy, E. (2013). Tailoring the automated construction of large-scale taxonomies using the web. *Language resources and evaluation*, 47(3):859–890.
- Levy, O., Remus, S., Biemann, C., and Dagan, I. (2015). Do Supervised Distributional Methods Really Learn Lexical Inference Relations? In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 970–976, Denver, CO, USA. Association for Computational Linguistics.
- Lin, D. and Pantel, P. (2001). Induction of Semantic Classes from Natural Language Text. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’01, pages 317–322, San Francisco, CA, USA. ACM.
- Meyer, C. M., Mieskes, M., Stab, C., and Gurevych, I. (2014). DKPro Agreement: An Open-Source Java Library for Measuring Inter-Rater Agreement. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: System Demonstrations*, pages 105–109, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.
- Mirkin, S., Dagan, I., and Geffet, M. (2006). Integrating pattern-based and distributional similarity methods for lexical entailment acquisition. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 579–586, Sydney, Australia. Association for Computational Linguistics.
- Navigli, R. and Ponzetto, S. P. (2012). BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.
- Necsulescu, S., Mendes, S., Jurgens, D., Bel, N., and Navigli, R. (2015). Reading Between the Lines: Overcoming Data Sparsity for Accurate Classification of Lexical Relationships. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 182–192, Denver, CO, USA. Association for Computational Linguistics.
- Neelakantan, A., Shankar, J., Passos, A., and McCallum, A. (2014). Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069, Doha, Qatar.
- Panchenko, A., Faralli, S., Ruppert, E., Remus, S., Naets, H., Fairon, C., Ponzetto, S. P., and Biemann, C. (2016). TAXI at SemEval-2016 Task 13: a Taxonomy Induction Method based on Lexico-Syntactic Patterns, Substrings and Focused Crawling. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1320–1327, San Diego, California, USA. Association for Computational Linguistics.
- Panchenko, A., Marten, F., Ruppert, E., Faralli, S., Ustalov, D., Ponzetto, S. P., and Biemann, C. (2017). Unsupervised, knowledge-free, and interpretable word sense disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*: System Demonstrations, pages 91–96, Copenhagen, Denmark, September. Association for Computational Linguistics.
- Pantel, P. and Lin, D. (2002). Discovering Word Senses from Text. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’02, pages 613–619, Edmonton, AB, Canada. ACM.
- Pantel, P. and Pennacchiotti, M. (2006). Espresso: Leveraging Generic Patterns for Automatically Harvesting Semantic Relations. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 113–120, Sydney, Australia. Association for Computational Linguistics.
- Pantel, P. and Ravichandran, D. (2004). Automatically Labeling Semantic Classes. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL’2004)*, pages 321–328, Boston, MA, USA. Association for Computational Linguistics.
- Randolph, J. J. (2005). Free-marginal multirater kappa (multirater k [free]): An alternative to fleiss’ fixed-marginal multirater kappa. *Online submission*.
- Richter, M., Quasthoff, U., Hallsteinsdóttir, E., and Biemann, C. (2006). Exploiting the Leipzig Corpora Collection. In *Proceedings of the Fifth Slovenian and First International Language Technologies Conference (IS-LTC)*, Ljubljana, Slovenia. European Language Resources Association (ELRA).
- Roller, S., Erk, K., and Boleda, G. (2014). Inclusive yet Selective: Supervised Distributional Hypernymy Detection. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1025–1036, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.
- Seitner, J., Bizer, C., Eckert, K., Faralli, S., Meusel, R., Paulheim, H., and Ponzetto, S. P. (2016). A Large DataBase of Hypernym Relations Extracted from the Web. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, LREC 2016, pages 360–367, Portorož, Slovenia. European Language Resources Association (ELRA).
- Shi, L. and Mihalcea, R. (2005). Putting Pieces Together: Combining FrameNet, VerbNet and WordNet for Robust Semantic Parsing. In *Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing 2005, pages 100–111, Mexico City, Mexico. Springer Berlin Heidelberg.
- Shwartz, V., Goldberg, Y., and Dagan, I. (2016). Improving Hypernymy Detection with an Integrated Path-based and Distributional Method. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2389–2398, Berlin, Germany, August. Association for Computational Linguistics.
- Snow, R., Jurafsky, D., and Ng, A. Y. (2004). Learning Syntactic Patterns for Automatic Hypernym Discovery.

- In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, NIPS'04, pages 1297–1304, Vancouver, BC, Canada. MIT Press.
- Steinberger, R., Pouliquen, B., Widiger, A., Ignat, C., Erjavec, T., Tufis, D., and Varga, D. (2006). The jrc-acquis: A multilingual aligned parallel corpus with 20+ languages. *arXiv preprint cs/0609058*.
- Steyvers, M. and Tenenbaum, J. B. (2005). The large-scale structure of semantic networks: Statistical analyses and a model of semantic growth. *Cognitive science*, 29(1):41–78.
- Tjong Kim Sang, E. (2007). Extracting Hypernym Pairs from the Web. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 165–168, Prague, Czech Republic. Association for Computational Linguistics.
- Ustalov, D., Arefyev, N., Biemann, C., and Panchenko, A. (2017a). Negative sampling improves hypernymy extraction based on projection learning. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 543–550, Valencia, Spain. Association for Computational Linguistics.
- Ustalov, D., Panchenko, A., and Biemann, C. (2017b). Watset: Automatic induction of synsets from a graph of synonyms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1579–1590, Vancouver, Canada. Association for Computational Linguistics.
- Velardi, P., Faralli, S., and Navigli, R. (2013). OntoLearn Reloaded: A Graph-Based Algorithm for Taxonomy Induction. *Computational Linguistics*, 39(3):665–707.
- Vylomova, E., Rimell, L., Cohn, T., and Baldwin, T. (2016). Take and Took, Gaggle and Goose, Book and Read: Evaluating the Utility of Vector Differences for Lexical Relation Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1671–1682, Berlin, Germany. Association for Computational Linguistics.
- Wandmacher, T. (2005). How semantic is Latent Semantic Analysis? In *Proceedings of RÉCITAL 2005*, pages 525–534, Dourdan, France.
- Weeds, J., Clarke, D., Reffin, J., Weir, D. J., and Keller, B. (2014). Learning to distinguish hypernyms and co-hyponyms. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2249–2259, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.
- Widdows, D. and Dorow, B. (2002). A graph model for unsupervised lexical acquisition. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.
- Zhou, G., Liu, Y., Liu, F., Zeng, D., and Zhao, J. (2013). Improving question retrieval in community question answering using world knowledge. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, pages 2239–2245, Beijing, China. AAAI Press.

A.14 Paper “Word Sense Disambiguation for 158 Languages using Word Embeddings Only”

V. Logacheva, D. Teslenko, A. Shelmanov, S. Remus, D. Ustalov, A. Kutuzov, E. Artemova, C. Biemann, S. P. Ponzetto, and A. Panchenko, “Word Sense Disambiguation for 158 Languages using Word Embeddings Only,” in *Proceedings of the Twelfth Language Resources and Evaluation Conference*, (Marseille, France), pp. 5943–5952, European Language Resources Association, May 2020

Permission to copy: the article is available in the public domain on the website at the following address: <https://aclanthology.org/2020.lrec-1.728>

Word Sense Disambiguation for 158 Languages using Word Embeddings Only

Varvara Logacheva¹, Denis Teslenko², Artem Shelmanov¹, Steffen Remus³,
Dmitry Ustalov^{4*}, Andrey Kutuzov⁵, Ekaterina Artemova⁶,
Chris Biemann³, Simone Paolo Ponzetto⁴, Alexander Panchenko¹

¹Skolkovo Institute of Science and Technology, Moscow, Russia

v.logacheva@skoltech.ru

²Ural Federal University, Yekaterinburg, Russia

³Universität Hamburg, Hamburg, Germany

⁴Universität Mannheim, Mannheim, Germany

⁵University of Oslo, Oslo, Norway

⁶Higher School of Economics, Moscow, Russia

Abstract

Disambiguation of word senses in context is easy for humans, but is a major challenge for automatic approaches. Sophisticated supervised and knowledge-based models were developed to solve this task. However, (i) the inherent Zipfian distribution of supervised training instances for a given word and/or (ii) the quality of linguistic knowledge representations motivate the development of completely unsupervised and knowledge-free approaches to word sense disambiguation (WSD). They are particularly useful for under-resourced languages which do not have any resources for building either supervised and/or knowledge-based models. In this paper, we present a method that takes as input a standard pre-trained word embedding model and induces a fully-fledged word sense inventory, which can be used for disambiguation in context. We use this method to induce a collection of sense inventories for 158 languages on the basis of the original pre-trained fastText word embeddings by Grave et al. (2018), enabling WSD in these languages. Models and system are available online.

Keywords: word sense induction, word sense disambiguation, word embeddings, sense embeddings, graph clustering

1. Introduction

There are many polysemous words in virtually any language. If not treated as such, they can hamper the performance of all semantic NLP tasks (Resnik, 2006). Therefore, the task of resolving the polysemy and choosing the most appropriate meaning of a word in context has been an important NLP task for a long time. It is usually referred to as Word Sense Disambiguation (WSD) and aims at assigning meaning to a word in context.

The majority of approaches to WSD are based on the use of knowledge bases, taxonomies, and other external manually built resources (Moro et al., 2014; Upadhyay et al., 2018). However, different senses of a polysemous word occur in very diverse contexts and can potentially be discriminated with their help. The fact that semantically related words occur in similar contexts, and diverse words do not share common contexts, is known as distributional hypothesis and underlies the technique of constructing word embeddings from unlabelled texts. The same intuition can be used to discriminate between different senses of individual words. There exist methods of training word embeddings that can detect polysemous words and assign them different vectors depending on their contexts (Athiwaratkun et al., 2018; Jain et al., 2019). Unfortunately, many widespread word embedding models, such as GloVe (Pennington et al., 2014), word2vec (Mikolov et al., 2013), fastText (Bojanowski et al., 2017), do not handle polysemous words. Words in these models are represented with single vectors, which were constructed from diverse sets of contexts corresponding to different senses. In such cases, their disam-

biguation needs knowledge-rich approaches.

We tackle this problem by suggesting a method of post-hoc unsupervised WSD. It does not require any external knowledge and can separate different senses of a polysemous word using only the information encoded in pre-trained word embeddings. We construct a semantic similarity graph for words and partition it into densely connected subgraphs. This partition allows for separating different senses of polysemous words. Thus, the only language resource we need is a large unlabelled text corpus used to train embeddings. This makes our method applicable to under-resourced languages. Moreover, while other methods of unsupervised WSD need to train embeddings from scratch, we perform retrofitting of sense vectors based on existing word embeddings.

We create a massively multilingual application for on-the-fly word sense disambiguation. When receiving a text, the system identifies its language and performs disambiguation of all the polysemous words in it based on pre-extracted word sense inventories. The system works for 158 languages, for which pre-trained fastText embeddings available (Grave et al., 2018).¹ The created inventories are based on these embeddings. To the best of our knowledge, our system is the only WSD system for the majority of the presented languages. Although it does not match the state of the art for resource-rich languages, it is fully unsupervised and can be used for virtually any language.

The contributions of our work are the following:

* Currently at Yandex.

¹The full list languages is available at fasttext.cc and includes English and 157 other languages for which embeddings were trained on a combination of Wikipedia and CommonCrawl texts.

- We release word sense inventories associated with fastText embeddings for 158 languages.
- We release a system that allows on-the-fly word sense disambiguation for 158 languages.
- We present `egvi` (Ego-Graph Vector Induction), a new algorithm of unsupervised word sense induction, which creates sense inventories based on pre-trained word vectors.

2. Related Work

There are two main scenarios for WSD: the supervised approach that leverages training corpora explicitly labelled for word sense, and the knowledge-based approach that derives sense representation from lexical resources, such as WordNet (Miller, 1995). In the supervised case WSD can be treated as a classification problem. Knowledge-based approaches construct sense embeddings, i.e. embeddings that separate various word senses.

SupWSD (Papandrea et al., 2017) is a state-of-the-art system for **supervised WSD**. It makes use of linear classifiers and a number of features such as POS tags, surrounding words, local collocations, word embeddings, and syntactic relations. GlossBERT model (Huang et al., 2019), which is another implementation of supervised WSD, achieves a significant improvement by leveraging gloss information. This model benefits from sentence-pair classification approach, introduced by Devlin et al. (2019) in their BERT contextualized embedding model. The input to the model consists of a context (a sentence which contains an ambiguous word) and a gloss (sense definition) from WordNet. The context-gloss pair is concatenated through a special token ([SEP]) and classified as positive or negative.

On the other hand, **sense embeddings** are an alternative to traditional word vector models such as word2vec, fastText or GloVe, which represent monosemous words well but fail for ambiguous words. Sense embeddings represent individual senses of polysemous words as separate vectors. They can be linked to an explicit inventory (Iacobacci et al., 2015) or induce a sense inventory from unlabelled data (Iacobacci and Navigli, 2019). LSTMEmbed (Iacobacci and Navigli, 2019) aims at learning sense embeddings linked to BabelNet (Navigli and Ponzetto, 2012), at the same time handling word ordering, and using pre-trained embeddings as an objective. Although it was tested only on English, the approach can be easily adapted to other languages present in BabelNet. However, manually labelled datasets as well as knowledge bases exist only for a small number of well-resourced languages. Thus, to disambiguate polysemous words in other languages one has to resort to fully unsupervised techniques.

The task of **Word Sense Induction** (WSI) can be seen as an unsupervised version of WSD. WSI aims at clustering word senses and does not require to map each cluster to a predefined sense. Instead of that, word sense inventories are induced automatically from the clusters, treating each cluster as a single sense of a word. WSI approaches fall into three main groups: context clustering, word ego-network clustering and synonyms (or substitute) clustering.

Context clustering approaches consist in creating vectors which characterise words’ contexts and clustering these

.Ruby, CRuby, CoffeeScript, Ember, Faye, Garnet, Gem, Groovy, Haskell, Hazel, JRuby, Jade, Jasmine, Josie, Jruby, Lottie, Millie, Oniguruma, Opal, Python, RUBY, Ruby., Ruby-like, Rabbitfoot, RubyMotion, Rails, Rubinius, Ruby-, Ruby-based, Ruby2, RubyGem, RubyGems, RubyInstaller, RubyOnRails, RubyRuby, RubySpec, Rubygems, Rubyist, Rubyists, Rubys, Sadie, Sapphire, Sypro, Violet, jRuby, ruby, rubyists

Table 1: Top nearest neighbours of the fastText vector of the word *Ruby* are clustered according to various senses of this word: **programming language**, **gem**, **first name**, **color**, but also its spelling variations (typeset in black color).

vectors. Here, the definition of context may vary from window-based context to latent topic-alike context. Afterwards, the resulting clusters are either used as senses directly (Kutuzov, 2018), or employed further to learn sense embeddings via Chinese Restaurant Process algorithm (Li and Jurafsky, 2015), AdaGram, a Bayesian extension of the Skip-Gram model (Bartunov et al., 2016), AutoSense, an extension of the LDA topic model (Amplayo et al., 2019), and other techniques.

Word ego-network clustering is applied to semantic graphs. The nodes of a semantic graph are words, and edges between them denote semantic relatedness which is usually evaluated with cosine similarity of the corresponding embeddings (Pelevina et al., 2016) or by PMI-like measures (Hope and Keller, 2013b). Word senses are induced via graph clustering algorithms, such as Chinese Whispers (Biemann, 2006) or MaxMax (Hope and Keller, 2013a). The technique suggested in our work belongs to this class of methods and is an extension of the method presented by Pelevina et al. (2016).

Synonyms and substitute clustering approaches create vectors which represent synonyms or substitutes of polysemous words. Such vectors are created using synonymy dictionaries (Ustalov et al., 2019) or context-dependent substitutes obtained from a language model (Amrami and Goldberg, 2018). Analogously to previously described techniques, word senses are induced by clustering these vectors.

3. Algorithm for Word Sense Induction

The majority of word vector models do not discriminate between multiple senses of individual words. However, a polysemous word can be identified via manual analysis of its nearest neighbours—they reflect different senses of the word. Table 1 shows manually sense-labelled most similar terms to the word *Ruby* according to the pre-trained fastText model (Grave et al., 2018). As it was suggested early by Widdows and Dorow (2002), the distributional properties of a word can be used to construct a graph of words that are semantically related to it, and if a word is polysemous, such graph can easily be partitioned into a number of densely connected subgraphs corresponding to different senses of this word. Our algorithm is based on the same principle.

3.1. SenseGram: A Baseline Graph-based Word Sense Induction Algorithm

SenseGram is the method proposed by Pelevina et al. (2016) that separates nearest neighbours to induce word senses and constructs sense embeddings for each sense. It starts by constructing an *ego-graph* (semantic graph centred at a particular word) of the word and its nearest neighbours. The edges between the words denote their semantic relatedness, e.g. the two nodes are joined with an edge if cosine similarity of the corresponding embeddings is higher than a pre-defined threshold. The resulting graph can be clustered into subgraphs which correspond to senses of the word.

The sense vectors are then constructed by averaging embeddings of words in each resulting cluster. In order to use these sense vectors for word sense disambiguation in text, the authors compute the probabilities of sense vectors of a word given its context or the similarity of the sense vectors to the context.

3.2. eegvi (Ego-Graph Vector Induction): A Novel Word Sense Induction Algorithm

Induction of Sense Inventories One of the downsides of the described above algorithm is noise in the generated graph, namely, unrelated words and wrong connections. They hamper the separation of the graph. Another weak point is the imbalance in the nearest neighbour list, when a large part of it is attributed to the most frequent sense, not sufficiently representing the other senses. This can lead to construction of incorrect sense vectors.

We suggest a more advanced procedure of graph construction that uses the interpretability of vector addition and subtraction operations in word embedding space (Mikolov et al., 2013) while the previous algorithm only relies on the list of nearest neighbours in word embedding space. The key innovation of our algorithm is the use of vector subtraction to find pairs of most dissimilar graph nodes and construct the graph only from the nodes included in such “anti-edges”. Thus, our algorithm is based on *graph-based* word sense induction, but it also relies on *vector-based* operations between word embeddings to perform filtering of graph nodes. Analogously to the work of Pelevina et al. (2016), we construct a semantic relatedness graph from a list of nearest neighbours, but we filter this list using the following procedure:

1. Extract a list $\mathcal{N} = \{w_1, w_2, \dots, w_N\}$ of N nearest neighbours for the target (ego) word vector w .
2. Compute a list $\Delta = \{\delta_1, \delta_2, \dots, \delta_N\}$ for each w_i in \mathcal{N} , where $\delta_i = w - w_i$. The vectors in δ contain the components of sense of w which are not related to the corresponding nearest neighbours from \mathcal{N} .
3. Compute a list $\bar{\mathcal{N}} = \{\bar{w}_1, \bar{w}_2, \dots, \bar{w}_{\bar{N}}\}$, such that \bar{w}_i is in the top nearest neighbours of δ_i in the embedding space. In other words, \bar{w}_i is a word which is the most similar to the target (ego) word w and least similar to its neighbour w_i . We refer to \bar{w}_i as an *anti-pair* of w_i . The set of N nearest neighbours and their anti-pairs form a set of *anti-edges* i.e. pairs of most dissimilar

nodes – those which should not be connected: $\bar{E} = \{(w_1, \bar{w}_1), (w_2, \bar{w}_2), \dots, (w_N, \bar{w}_N)\}$.

To clarify this, consider the target (ego) word $w = \text{python}$, its top similar term $w_1 = \text{Java}$ and the resulting anti-pair $\bar{w}_1 = \text{snake}$ which is the top related term of $\delta_1 = w - w_1$. Together they form an anti-edge $(w_1, \bar{w}_1) = (\text{Java}, \text{snake})$ composed of a pair of semantically dissimilar terms.

4. Construct V , the set of vertices of our semantic graph $G = (V, E)$ from the list of anti-edges \bar{E} , with the following recurrent procedure: $V = V \cup \{w_i, \bar{w}_i : w_i \in \mathcal{N}, \bar{w}_i \in \mathcal{N}\}$, i.e. we add a word from the list of nearest neighbours *and* its anti-pair only if both of them are nearest neighbours of the original word w . We do not add w 's nearest neighbours if their anti-pairs do not belong to \mathcal{N} . Thus, we add only words which can help discriminating between different senses of w .
5. Construct the set of edges E as follows. For each $w_i \in \mathcal{N}$ we extract a set of its K nearest neighbours $\mathcal{N}'_i = \{u_1, u_2, \dots, u_K\}$ and define $E = \{(w_i, u_j) : w_i \in V, u_j \in V, u_j \in \mathcal{N}'_i, u_j \neq \bar{w}_i\}$. In other words, we remove edges between a word w_i and its nearest neighbour u_j if u_j is also its anti-pair. According to our hypothesis, w_i and \bar{w}_i belong to different senses of w , so they should not be connected (i.e. we never add anti-edges into E). Therefore, we consider any connection between them as noise and remove it.

Note that N (the number of nearest neighbours for the target word w) and K (the number of nearest neighbours of w_{ci}) do not have to match. The difference between these parameters is the following. N defines how many words will be considered for the construction of ego-graph. On the other hand, K defines the degree of relatedness between words in the ego-graph — if $K = 50$, then we will connect vertices w and u with an edge only if u is in the list of 50 nearest neighbours of w . Increasing K increases the graph connectivity and leads to lower granularity of senses.

According to our hypothesis, nearest neighbours of w are grouped into clusters in the vector space, and each of the clusters corresponds to a sense of w . The described vertices selection procedure allows picking the most representative members of these clusters which are better at discriminating between the clusters. In addition to that, it helps dealing with the cases when one of the clusters is over-represented in the nearest neighbour list. In this case, many elements of such a cluster are not added to V because their anti-pairs fall outside the nearest neighbour list. This also improves the quality of clustering.

After the graph construction, the clustering is performed using the Chinese Whispers algorithm (Biemann, 2006). This is a bottom-up clustering procedure that does not require to pre-define the number of clusters, so it can correctly process polysemous words with varying numbers of senses as well as unambiguous words.

Figure 1 shows an example of the resulting pruned graph of for the word *Ruby* for $N = 50$ nearest neighbours in terms of the fastText cosine similarity. In contrast to the baseline method by (Pelevina et al., 2016) where all 50 terms are

clustered, in the method presented in this section we sparsify the graph by removing 13 nodes which were not in the set of the “anti-edges” i.e. pairs of most dissimilar terms out of these 50 neighbours. Examples of anti-edges i.e. pairs of most dissimilar terms for this graph include: (*Haskell*, *Sapphire*), (*Garnet*, *Rails*), (*Opal*, *Rubyist*), (*Hazel*, *Ruby-OnRails*), and (*Coffeescript*, *Opal*).

Labelling of Induced Senses We label each word cluster representing a sense to make them and the WSD results interpretable by humans. Prior systems used hypernyms to label the clusters (Ruppert et al., 2015; Panchenko et al., 2017), e.g. “animal” in the “python (animal)”. However, neither hypernyms nor rules for their automatic extraction are available for all 158 languages. Therefore, we use a simpler method to select a keyword which would help to interpret each cluster. For each graph node $v \in V$ we count the number of anti-edges it belongs to: $count(v) = |\{(w_i, \bar{w}_i) : (w_i, \bar{w}_i) \in \bar{E} \wedge (v = w_i \vee v = \bar{w}_i)\}|$. A graph clustering yields a partition of V into n clusters: $V = \{V_1, V_2, \dots, V_n\}$. For each cluster V_i we define a keyword w_i^{key} as the word with the largest number of anti-edges $count(\cdot)$ among words in this cluster.

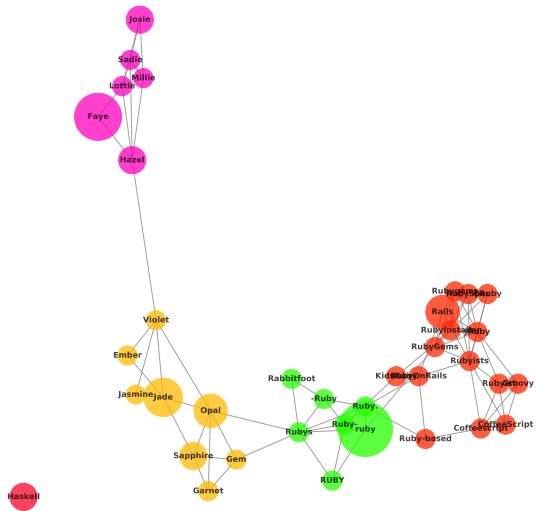


Figure 1: The graph of nearest neighbours of the word *Ruby* can be separated according several senses: programming languages, female names, gems, as well as a cluster of different spellings of the word *Ruby*.

Word Sense Disambiguation We use keywords defined above to obtain vector representations of senses. In particular, we simply use word embedding of the keyword w_i^{key} as a sense representation s_i of the target word w to avoid explicit computation of sense embeddings like in (Pelevina et al., 2016). Given a sentence $\{w_1, w_2, \dots, w_j, w, w_{j+1}, \dots, w_n\}$ represented as a matrix of word vectors, we define the context of the target word w as $\mathbf{c}_w = \frac{\sum_{j=1}^n w_j}{n}$. Then, we define the most appropriate sense \hat{s} as the sense with the highest cosine similarity to

the embedding of the word’s context:

$$\hat{s} = \arg \max_{s_i} \frac{\mathbf{c}_w \cdot \mathbf{s}_i}{\|\mathbf{c}_w\| \cdot \|\mathbf{s}_i\|}.$$

4. System Design

We release a system for on-the-fly WSD for 158 languages. Given textual input, it identifies polysemous words and retrieves senses that are the most appropriate in the context.

4.1. Construction of Sense Inventories

To build word sense inventories (sense vectors) for 158 languages, we utilised GPU-accelerated routines for search of similar vectors implemented in Faiss library (Johnson et al., 2019). The search of nearest neighbours takes substantial time, therefore, acceleration with GPUs helps to significantly reduce the word sense construction time. To further speed up the process, we keep all intermediate results in memory, which results in substantial RAM consumption of up to 200 Gb.

The construction of word senses for all of the 158 languages takes a lot of computational resources and imposes high requirements to the hardware. For calculations, we use in parallel 10–20 nodes of the Zhores cluster (Zacharov et al., 2019) empowered with Nvidia Tesla V100 graphic cards. For each of the languages, we construct inventories based on 50, 100, and 200 neighbours for 100,000 most frequent words. The vocabulary was limited in order to make the computation time feasible. The construction of inventories for one language takes up to 10 hours, with 6.5 hours on average. Building the inventories for all languages took more than 1,000 hours of GPU-accelerated computations.

We release the constructed sense inventories for all the available languages. They contain all the necessary information for using them in the proposed WSD system or in other downstream tasks.

4.2. Word Sense Disambiguation System

The first text pre-processing step is language identification, for which we use the fastText language identification models by Bojanowski et al. (2017). Then the input is tokenised. For languages which use Latin, Cyrillic, Hebrew, or Greek scripts, we employ the Europarl tokeniser.² For Chinese, we use the Stanford Word Segemer (Tseng et al., 2005). For Japanese, we use Mecab (Kudo, 2006). We tokenise Vietnamese with UETsegemer (Nguyen and Le, 2016). All other languages are processed with the ICU tokeniser, as implemented in the PyICU project.³ After the tokenisation, the system analyses all the input words with pre-extracted sense inventories and defines the most appropriate sense for polysemous words.

Figure 2 shows the interface of the system. It has a textual input form. The automatically identified language of text is shown above. A click on any of the words displays a prompt (shown in black) with the most appropriate sense of a word in the specified context and the confidence score. In the given example, the word *Jaguar* is correctly identified as a car brand. This system is based on the system

²<https://www.statmt.org/europarl>

³<https://pypi.org/project/PyICU>

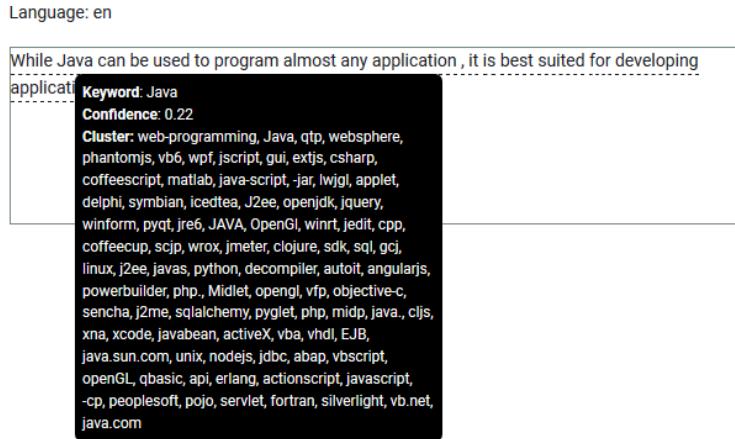


Figure 2: Interface of our WSD module with examples for the English language. Given a sentence, it identifies polysemous words and retrieves the most appropriate sense (labelled by the centroid word of a corresponding cluster).

by Ustalov et al. (2018), extending it with a back-end for multiple languages, language detection, and sense browsing capabilities.

5. Evaluation

We first evaluate our converted embedding models on multi-language lexical similarity and relatedness tasks, as a sanity check, to make sure the word sense induction process did not hurt the general performance of the embeddings. Then, we test the sense embeddings on WSD task.

5.1. Lexical Similarity and Relatedness

Experimental Setup We use the SemR-11 datasets⁴ (Barzegar et al., 2018), which contain word pairs with manually assigned similarity scores from 0 (words are not related) to 10 (words are fully interchangeable) for 12 languages: English (en), Arabic (ar), German (de), Spanish (es), Farsi (fa), French (fr), Italian (it), Dutch (nl), Portuguese (pt), Russian (ru), Swedish (sv), Chinese (zh). The task is to assign relatedness scores to these pairs so that the ranking of the pairs by this score is close to the ranking defined by the oracle score. The performance is measured with Pearson correlation of the rankings. Since one word can have several different senses in our setup, we follow Remus and Biemann (2018) and define the relatedness score for a pair of words as the **maximum cosine similarity** between any of their sense vectors.

We extract the sense inventories from fastText embedding vectors. We set $N = K$ for all our experiments, i.e. the number of vertices in the graph and the maximum number of vertices’ nearest neighbours match. We conduct experiments with $N = K$ set to 50, 100, and 200. For each cluster V_i we create a sense vector s_i by averaging vectors that belong to this cluster. We rely on the methodology of (Remus and Biemann, 2018) shifting the generated sense vector to the direction of the original word vector: $s_i = \lambda w + (1 - \lambda) \frac{1}{n} \sum_{u \in V_i} \cos(w, u) \cdot u$

⁴<https://github.com/Lambda-3/Gold-Standards/tree/master/SemR-11>

where, $\lambda \in [0, 1]$, w is the embedding of the original word, $\cos(w, u)$ is the cosine similarity between w and u , and $n = |V_i|$. By introducing the linear combination of w and $u \in V_i$ we enforce the similarity of sense vectors to the original word important for this task. In addition to that, we weight u by their similarity to the original word, so that more similar neighbours contribute more to the sense vector. The shifting parameter λ is set to 0.5, following Remus and Biemann (2018).

A fastText model is able to generate a vector for each word even if it is not represented in the vocabulary, due to the use of subword information. However, our system cannot assemble sense vectors for out-of-vocabulary words, for such words it returns their original fastText vector. Still, the coverage of the benchmark datasets by our vocabulary is at least 85% and approaches 100% for some languages, so we do not have to resort to this back-off strategy very often. We use the original fastText vectors as a **baseline**. In this case, we compute the relatedness scores of the two words as a cosine similarity of their vectors.

Discussion of Results We compute the relatedness scores for all benchmark datasets using our sense vectors and compare them to cosine similarity scores of original fastText vectors. The results vary for different languages. Figure 3 shows the change in Pearson correlation score when switching from the baseline fastText embeddings to our sense vectors. The new vectors significantly improve the relatedness detection for German, Farsi, Russian, and Chinese, whereas for Italian, Dutch, and Swedish the score slightly falls behind the baseline. For other languages, the performance of sense vectors is on par with regular fastText.

5.2. Word Sense Disambiguation

The purpose of our sense vectors is disambiguation of polysemous words. Therefore, we test the inventories constructed with egvi on the Task 13 of SemEval-2013 — Word Sense Induction (Jurgens and Klaptidis, 2013). The task is to identify the different senses of a target word in context in a fully unsupervised manner.

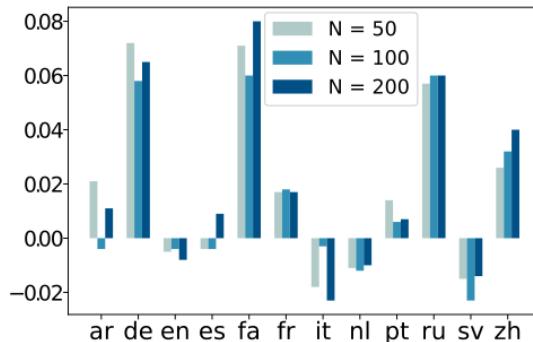


Figure 3: Absolute improvement of Pearson correlation scores of our embeddings compared to fastText. This is the averaged difference of the scores for all word similarity benchmarks.

Experimental Setup The dataset consists of a set of polysemous words: 20 nouns, 20 verbs, and 10 adjectives and specifies 20 to 100 contexts per word, with the total of 4,664 contexts, drawn from the Open American National Corpus. Given a set of contexts of a polysemous word, the participants of the competition had to divide them into clusters by sense of the word. The contexts are manually labelled with WordNet senses of the target words, the gold standard clustering is generated from this labelling.

The task allows two setups: *graded* WSI where participants can submit multiple senses per word and provide the probability of each sense in a particular context, and *non-graded* WSI where a model determines a single sense for a word in context. In our experiments we performed *non-graded* WSI. We considered the most suitable sense as the one with the highest cosine similarity with embeddings of the context, as described in Section 3.2.

The performance of WSI models is measured with three metrics that require mapping of sense inventories (Jaccard Index, Kendall’s τ , and WNDCG) and two cluster comparison metrics (Fuzzy NMI and Fuzzy B-Cubed).

Discussion of Results We compare our model with the models that participated in the task, the baseline ego-graph clustering model by Pelevina et al. (2016), and AdaGram (Bartunov et al., 2016), a method that learns sense embeddings based on a Bayesian extension of the Skip-gram model. Besides that, we provide the scores of the simple **baselines** originally used in the task: assigning one sense to all words, assigning the most frequent sense to all words, and considering each context as expressing a different sense. The evaluation of our model was performed using the open source `context-eval` tool.⁵

Table 2 shows the performance of these models on the SemEval dataset. Due to space constraints, we only report the scores of the best-performing SemEval participants, please refer to Jurgens and Klapaftis (2013) for the full results. The performance of AdaGram and SenseGram models is reported according to Pelevina et al. (2016).

The table shows that the performance of `egvi` is simi-

lar to state-of-the-art word sense disambiguation and word sense induction models. In particular, we can see that it outperforms SenseGram on the majority of metrics. We should note that this comparison is not fully rigorous, because SenseGram induces sense inventories from word2vec as opposed to fastText vectors used in our work.

5.3. Analysis

In order to see how the separation of word contexts that we perform corresponds to actual senses of polysemous words, we visualise ego-graphs produced by our method. Figure 1 shows the nearest neighbours clustering for the word *Ruby*, which divides the graph into five senses: *Ruby-related programming tools*, e.g. RubyOnRails (orange cluster), *female names*, e.g. Josie (magenta cluster), *gems*, e.g. Sapphire (yellow cluster), *programming languages in general*, e.g. Haskell (red cluster). Besides, this is typical for fastText embeddings featuring sub-string similarity, one can observe a cluster of different spelling of the word *Ruby* in green. Analogously, the word *python* (see Figure 4) is divided into the senses of *animals*, e.g. crocodile (yellow cluster), *programming languages*, e.g. perl5 (magenta cluster), and *conference*, e.g. pycon (red cluster).

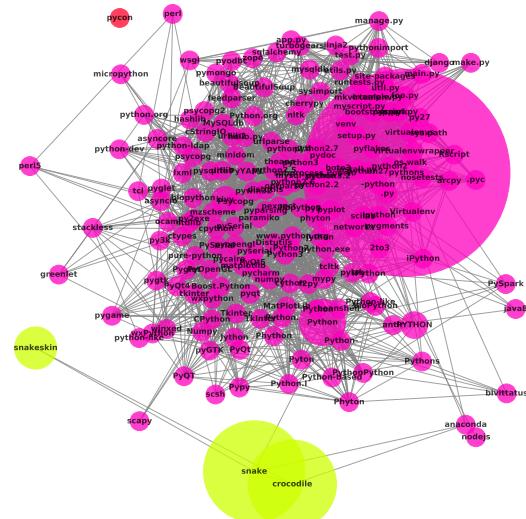


Figure 4: Ego-graph for a polysemous word *python* which is clustered into senses *snake* (yellow), *programming language* (magenta), and *conference* (red). Node size denotes word importance with the largest node in the cluster being used as a keyword to interpret an induced word sense.

In addition, we show a qualitative analysis of senses of *mouse* and *apple*. Table 4 shows nearest neighbours of the original words separated into clusters (labels for clusters were assigned manually). These inventories demonstrate clear separation of different senses, although it can be too fine-grained. For example, the first and the second cluster for *mouse* both refer to computer mouse, but the first one addresses the different types of computer mice, and the second one is used in the context of mouse actions. Similarly, we see that *iphone* and *macbook* are sep-

⁵<https://github.com/uhh-lt/context-eval>

Model	Supervised Evaluation			Clustering Evaluation	
	Jacc. Ind.	Kendall's τ	WNDCG	F.NMI	F.B-Cubed
Baselines					
One sense for all	0.192	0.609	0.288	0.000	0.631
One sense per instance	0.000	0.000	0.000	0.071	0.000
Most Frequent Sense	0.455	0.465	0.339	–	–
SemEval-2013 participants					
AI-KU (base)	0.197	0.620	0.387	0.065	0.390
AI-KU (remove5-add1000)	0.244	0.642	0.332	0.039	0.451
Unimelb (50k)	0.213	0.620	0.371	0.060	0.483
Sense embeddings					
AdaGram, $\alpha = 0.05$, 100 dim. vectors	0.274	0.644	0.318	0.058	0.470
SenseGram (word2vec)	0.197	0.615	0.291	0.011	0.615
egvi (fastText, K=200)	0.229	0.625	0.300	0.035	0.541

Table 2: WSD performance on the SemEval-2013 Task 13 dataset for the English language.

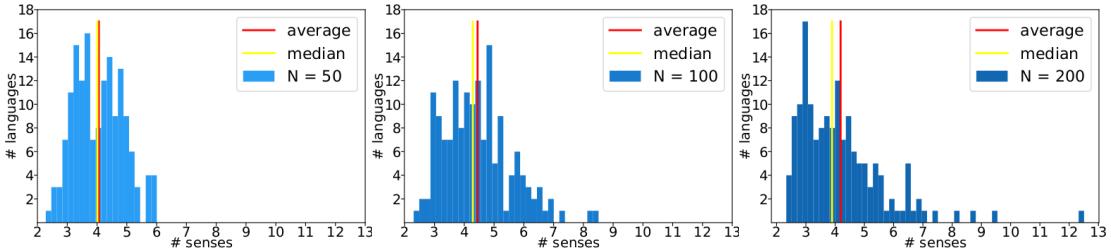


Figure 5: Distribution of the number of senses per word in the generated inventories for all 158 languages for the number of neighbours set to: $N \in \{50, 100, 200\}$, $K \in \{50, 100, 200\}$ with $N = K$.

erated into two clusters. Interestingly, fastText handles typos, code-switching, and emojis by correctly associating all non-standard variants to the word they refer, and our method is able to cluster them appropriately.

Both inventories were produced with $K = 200$, which ensures stronger connectivity of graph. However, we see that this setting still produces too many clusters. We computed the average numbers of clusters produced by our model with $K = 200$ for words from the word relatedness datasets and compared these numbers with the number of senses in WordNet for English and RuWordNet (Loukachevitch and Dobrov, 2014) for Russian (see Table 3). We can see that the number of senses extracted by our method is consistently higher than the real number of senses.

We also compute the average number of senses per word for all the languages and different values of K (see Figure 5). The average across languages does not change much as we increase K . However, for larger K the average exceed the median value, indicating that more languages have lower number of senses per word. At the same time, while at smaller K the maximum average number of senses per word does not exceed 6, larger values of K produce outliers, e.g. English with 12.5 senses.

Notably, there are no languages with an average number of senses less than 2, while numbers on English and Russian WordNets are considerably lower. This confirms that our method systematically over-generates senses. The presence of outliers shows that this effect cannot be eliminated by

further increasing K , because the i -th nearest neighbour of a word for $i > 200$ can be only remotely related to this word, even if the word is rare. Thus, our sense clustering algorithm needs a method of merging spurious senses.

	mc	rg	simlex	ws353	total
English					
inventory	9.8	9.8	12.6	11.3	12.5
WordNet	3.6	3.7	6.5	5.5	1.23 (nouns) 2.16 (verbs)
Russian					
inventory	1.8	2.0	–	2.2	2.97
RuWordNet	1.4	1.4	–	1.8	1.12 (nouns) 1.33 (verbs)

Table 3: Average number of senses for words from SemR-11 dataset in our inventory and in WordNet for English and ruWordNet for Russian. The rightmost column gives the average number of senses in the inventories and WordNets.

6. Conclusions and Future Work

We present **egvi**, a new algorithm for word sense induction based on graph clustering that is fully unsupervised and relies on graph operations between word vectors. We

Label	Nearest neighbours
MOUSE	
computer mouse types	touch-pad, logitec, scrollwheel, mouses , mouse.It, mouse.The, joystick, trackpads, nano-receiver, 800DPI, nony, track-pad, M325, keyboard-controlled, Intellipoint, MouseI, intellimouse, Swiftpoint, Evoluent, 800dpi, moused, game-pad, steelseries, ErgoMotion, IntelliMouse, <...>
computer mouse actions	Ctrl-Alt, right-mouse, cursor, left-clicks, spacebar, rnUse, mouseclick, click , mousepointer, keystroke, cusr, mousewheel, MouseMove, mousebutton, leftclick, click-dragging, mouse-button, cursor., arrow-key, double-clicks, mouse-down, ungrab, mouseX, arrow-keys, right-button, <...>
rodent	Rodent, rodent, mousehole, rats, mice , mice-, hamster, SOD1G93A, meeses, mice.The, PDAPP, hedgehog, Maukie, rTg4510, mousey, meees, rodents, cat, White-footed, rat, Mice, <...>
keyboard	keyborad, keybard, keybord, keyboardThe, keyboard, keyboar, Keyboard , keboard, keyboardI, keyb, keyboard.This, keybaord, keyboard
medical	SENCAR, mTERT, mouse-specific
Latin	Apodemus, Micormys
Latin	Akodon
APPLE	
iphone	mobileme, idevice, carplay, iphones, icloud, iwatch, ios5, ipod, iphone , android, ifans, iphone.I, iphone4, iphone5s, idevices, ipad, ios, ipad., iphone5, iphone., ios7
fruit	apples , apple-producing, Honeycrisp, apple-y, Macouns, apple-growing, pear, apple-pear, Gravensteins, apple-like, Apples, Honeycrisps, apple-related, Borkh, Braeburns, Starkrimson, Apples-, SweeTango, Elstar
macbook	macbook, macbookpro, macbookair, imac, ibooks, tuaw , osx, macintosh, imacs, apple.com, applestore, Tagsapple, stevejobs, applecare
fruit, typos pinklady	blackerry, blackberry, blueberry, aplle, cedar, apple.The , apple.I, aple, appple, calvados, pie.It,
tokenisation issues, typos	Apple.This, AMAApple, it.Apple, too.Apple, AppleApple, up.Apple , AppleA, Apline, Apple..Apple
Apple criticism	anti-apple, Aple, Crapple, isheep, iDiots, crapple, Appple, iCrap, non-apple
Bible	Adam
cooking	caramel-dipped
iphone	earpod
Russian	яблоко [Russian: “apple”]
emoji	[apple emoji]

Table 4: Clustering of senses for words *mouse* and *apple* produced by our method. Cluster labels in this table were assigned manually for illustrative purposes. For on-the-fly disambiguation we use centroid words in clusters as sense labels (shown here in **bold**).

apply this algorithm to a large collection of pre-trained fastText word embeddings, releasing sense inventories for 158 languages.⁶ These inventories contain all the necessary information for constructing sense vectors and using them in downstream tasks. The sense vectors for polysemous words can be directly retrofitted with the pre-trained word embeddings and do not need any external resources. As one application of these multilingual sense inventories, we present a multilingual word sense disambiguation system that performs unsupervised and knowledge-free WSD for 158 languages without the use of any dictionary or sense-labelled corpus.

The evaluation of quality of the produced sense inventories is performed on multilingual word similarity benchmarks, showing that our sense vectors improve the scores

compared to non-disambiguated word embeddings. Therefore, our system in its present state can improve WSD and downstream tasks for languages where knowledge bases, taxonomies, and annotated corpora are not available and supervised WSD models cannot be trained.

A promising direction for future work is combining distributional information from the induced sense inventories with lexical knowledge bases to improve WSD performance. Besides, we encourage the use of the produced word sense inventories in other downstream tasks.

7. Acknowledgements

We acknowledge the support of the Deutsche Forschungsgemeinschaft (DFG) foundation under the “JOIN-T 2” and “ACQuA” projects. Ekaterina Artemova was supported by the framework of the HSE University Basic Research Program and Russian Academic Excellence Project “5-100”.

⁶Links to the produced datasets, online demo, and source codes are available at: <http://uhh-lt.github.io/158>.

8. Bibliographical References

- Amplayo, R. K., Hwang, S.-w., and Song, M. (2019). AutoSense Model for Word Sense Induction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6212–6219, Honolulu, HI, USA. Association for the Advancement of Artificial Intelligence (AAAI).
- Amrami, A. and Goldberg, Y. (2018). Word Sense Induction with Neural biLM and Symmetric Patterns. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2018, pages 4860–4867, Brussels, Belgium. Association for Computational Linguistics.
- Athiwaratkun, B., Wilson, A., and Anandkumar, A. (2018). Probabilistic FastText for Multi-Sense Word Embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2018, pages 1–11, Melbourne, VIC, Australia. Association for Computational Linguistics.
- Bartunov, S., Kondrashkin, D., Osokin, A., and Vetrov, D. P. (2016). Breaking Sticks and Ambiguities with Adaptive Skip-gram. *Journal of Machine Learning Research*, 51:130–138.
- Barzegar, S., Davis, B., Zarrouk, M., Handschuh, S., and Freitas, A. (2018). SemR-11: A Multi-Lingual Gold-Standard for Semantic Similarity and Relatedness for Eleven Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 3912–3916, Miyazaki, Japan. European Language Resources Association (ELRA).
- Biemann, C. (2006). Chinese Whispers: An Efficient Graph Clustering Algorithm and Its Application to Natural Language Processing Problems. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*, TextGraphs-1, pages 73–80, New York, NY, USA. Association for Computational Linguistics.
- Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, NAACL-HLT 2019, pages 4171–4186, Minneapolis, MN, USA. Association for Computational Linguistics.
- Grave, E., Bojanowski, P., Gupta, P., Joulin, A., and Mikolov, T. (2018). Learning Word Vectors for 157 Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 3483–3487, Miyazaki, Japan. European Language Resources Association (ELRA).
- Hope, D. and Keller, B. (2013a). MaxMax: A Graph-Based Soft Clustering Algorithm Applied to Word Sense Induction. In *Computational Linguistics and Intelligent Text Processing, 14th International Conference, CICLING 2013, Samos, Greece, March 24–30, 2013, Proceedings, Part I*, volume 7816 of *Lecture Notes in Computer Science*, pages 368–381. Springer Berlin Heidelberg, Berlin and Heidelberg, Germany.
- Hope, D. and Keller, B. (2013b). UoS: A Graph-Based System for Graded Word Sense Induction. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 689–694, Atlanta, GA, USA. Association for Computational Linguistics.
- Huang, L., Sun, C., Qiu, X., and Huang, X. (2019). GlossBERT: BERT for Word Sense Disambiguation with Gloss Knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3509–3514, Hong Kong. Association for Computational Linguistics.
- Iacobacci, I. and Navigli, R. (2019). LSTMEmbed: Learning Word and Sense Representations from a Large Semantically Annotated Corpus with Long Short-Term Memories. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, ACL 2019, pages 1685–1695, Florence, Italy. Association for Computational Linguistics.
- Iacobacci, I., Pilehvar, M. T., and Navigli, R. (2015). Sensembed: Learning sense embeddings for word and relational similarity. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, ACL-IJCNLP 2015, pages 95–105, Beijing, China. Association for Computational Linguistics.
- Jain, S., Bodapati, S. B., Nallapati, R., and Anandkumar, A. (2019). Multi Sense Embeddings from Topic Models. In *Proceedings of the 3rd International Conference on Natural Language and Speech Processing*, ICNLSP 2019, pages 34–41, Trento, Italy. Association for Computational Linguistics.
- Johnson, J., Douze, M., and Jégou, H. (2019). Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*.
- Jurgens, D. and Klapaftis, I. (2013). SemEval-2013 task 13: Word sense induction for graded and non-graded senses. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 290–299, Atlanta, GA, USA, June. Association for Computational Linguistics.
- Kudo, T. (2006). MeCab: Yet Another Part-of-Speech and Morphological Analyzer. <http://mecab.sourceforge.jp/>.
- Kutuzov, A. (2018). Russian word sense induction by clustering averaged word embeddings. In *Komp'yuternaya Lingvistika i Intellektual'nye Tekhnologii: Dialog conference*, pages 391–403, Moscow, Russia.
- Li, J. and Jurafsky, D. (2015). Do Multi-Sense Embed-

- dings Improve Natural Language Understanding? In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2015, pages 1722–1732, Lisbon, Portugal. Association for Computational Linguistics.
- Loukachevitch, N. and Dobrov, B. (2014). RuThes Linguistic Ontology vs. Russian Wordnets. In *Proceedings of the Seventh Global Wordnet Conference*, GWC 2014, pages 154–162, Tartu, Estonia. University of Tartu Press.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, NIPS 2013, pages 3111–3119. Curran Associates, Inc., Harrahs and Harveys, NV, USA.
- Miller, G. A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41.
- Moro, A., Cecconi, F., and Navigli, R. (2014). Multilingual Word Sense Disambiguation and Entity Linking for Everybody. In *Proceedings of the ISWC 2014 Posters & Demonstrations Track*, pages 25–28, Riva del Garda, Italy.
- Navigli, R. and Ponzetto, S. P. (2012). BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.
- Nguyen, T.-P. and Le, A.-C. (2016). A hybrid approach to vietnamese word segmentation. In *2016 IEEE RIVF International Conference on Computing Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)*, pages 114–119, Hanoi, Vietnam. IEEE.
- Panchenko, A., Marten, F., Ruppert, E., Faralli, S., Ustalov, D., Ponzetto, S. P., and Biemann, C. (2017). Unsupervised, knowledge-free, and interpretable word sense disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 91–96, Copenhagen, Denmark, September. Association for Computational Linguistics.
- Papandrea, S., Raganato, A., and Delli Bovi, C. (2017). SupWSD: A Flexible Toolkit for Supervised Word Sense Disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, EMNLP 2017, pages 103–108, Copenhagen, Denmark. Association for Computational Linguistics.
- Pelevina, M., Arefiev, N., Biemann, C., and Panchenko, A. (2016). Making Sense of Word Embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, RePL4NLP, pages 174–183, Berlin, Germany. Association for Computational Linguistics.
- Pennington, J., Socher, R., and Manning, C. (2014). GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2014, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.
- Remus, S. and Biemann, C. (2018). Retrofitting Word Representations for Unsupervised Sense Aware Word Similarities. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 1035–1041, Miyazaki, Japan. European Language Resources Association (ELRA).
- Resnik, P. (2006). WSD in NLP Applications. In *Word Sense Disambiguation*, pages 299–337. Springer.
- Ruppert, E., Kaufmann, M., Riedl, M., and Biemann, C. (2015). JoBimViz: A web-based visualization for graph-based distributional semantic models. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 103–108, Beijing, China, July. Association for Computational Linguistics and The Asian Federation of Natural Language Processing.
- Tseng, H., Chang, P., Andrew, G., Jurafsky, D., and Manning, C. (2005). A Conditional Random Field Word Segmenter for SIGHAN Bakeoff 2005. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, pages 168–171.
- Upadhyay, S., Gupta, N., and Roth, D. (2018). Joint Multilingual Supervision for Cross-lingual Entity Linking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2018, pages 2486–2495, Brussels, Belgium. Association for Computational Linguistics.
- Ustalov, D., Teslenko, D., Panchenko, A., Chersnoskutov, M., Biemann, C., and Ponzetto, S. P. (2018). An Unsupervised Word Sense Disambiguation System for Under-Resourced Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 1018–1022, Miyazaki, Japan. European Language Resources Association (ELRA).
- Ustalov, D., Panchenko, A., Biemann, C., and Ponzetto, S. P. (2019). WATSET: Local-Global Graph Clustering with Applications in Sense and Frame Induction. *Computational Linguistics*, 45(3):423–479.
- Widdows, D. and Dorow, B. (2002). A Graph Model for Unsupervised Lexical Acquisition. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Taipei, Taiwan. Association for Computational Linguistics.
- Zacharov, I., Arslanov, R., Gunin, M., Stefonishin, D., Bykov, A., Pavlov, S., Panarin, O., Maliutin, A., Rykovyanov, S., and Fedorov, M. (2019). “Zhores” — Petaflops supercomputer for data-driven modeling, machine learning and artificial intelligence installed in Skolkovo Institute of Science and Technology. *Open Engineering*, 9(1):512–520.