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

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 MaxMax (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, Panchenko 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 Pachenko 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 $\textsc{Cluster} : (V, E) \to C$ 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.

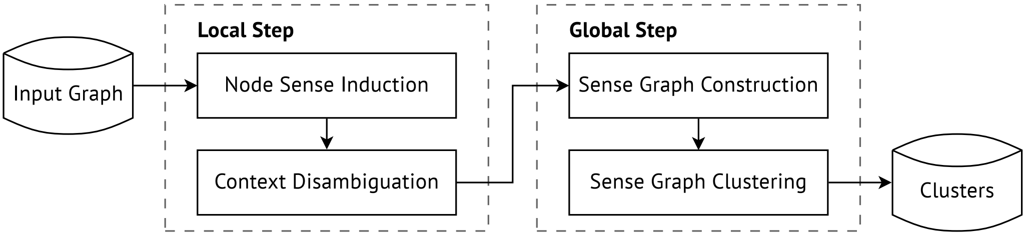


Figure 1
The outline of the WATSET algorithm showing the *local* step of word sense induction and context disambiguation, and the *global* step of sense graph constructing and clustering.

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 $\text{sim} : (\text{ctx}(a), \text{ctx}(b)) \to \mathbb{R}, \forall\, \text{ctx}(a), \text{ctx}(b) \subseteq V$.
Output: clusters $C$.
for all $u \in V$ do                                              ▷ Local Step: Sense Induction
    $\text{senses}(u) \leftarrow \emptyset$
    $V_u \leftarrow \{v \in V : \{u, v\} \in E\}$                 ▷ Note that $u \notin V_u$
    $E_u \leftarrow \{\{v, w\} \in E : v, w \in V_u\}$
    $G_u \leftarrow (V_u, E_u)$
    $C_u \leftarrow \text{Cluster}_{\text{Local}}(G_u)$           ▷ Cluster the open neighborhood of $u$
    for all $C_u^i \in C_u$ do
        $\text{ctx}(u^i) \leftarrow C_u^i$
        $\text{senses}(u) \leftarrow \text{senses}(u) \cup \{u^i\}$
$\mathcal{V} \leftarrow \bigcup_{u \in V} \text{senses}(u)$       ▷ Global Step: Sense Graph Nodes
for all $\hat{u} \in \mathcal{V}$ do                              ▷ Local Step: Context Disambiguation
    $\widehat{\text{ctx}}(\hat{u}) \leftarrow \emptyset$
    for all $v \in \text{ctx}(\hat{u})$ do
        $\hat{v} \leftarrow \arg\max_{v' \in \text{senses}(v)} \text{sim}(\text{ctx}(\hat{u}) \cup \{u\}, \text{ctx}(v'))$    ▷ $\hat{u}$ is a sense of $u \in V$
        $\widehat{\text{ctx}}(\hat{u}) \leftarrow \widehat{\text{ctx}}(\hat{u}) \cup \{\hat{v}\}$
$\mathcal{E} \leftarrow \{\{\hat{u}, \hat{v}\} \in \mathcal{V}^2 : \hat{v} \in \widehat{\text{ctx}}(\hat{u})\}$    ▷ Global Step: Sense Graph Edges
$\mathcal{G} \leftarrow (\mathcal{V}, \mathcal{E})$               ▷ Global Step: Sense Graph Construction
$\mathcal{C} \leftarrow \text{Cluster}_{\text{Global}}(\mathcal{G})$    ▷ Global Step: Sense Graph Clustering
$C \leftarrow \{\{u \in V : \hat{u} \in \mathcal{C}^i\} \subseteq V : \mathcal{C}^i \in \mathcal{C}\}$    ▷ Remove the sense labels
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.

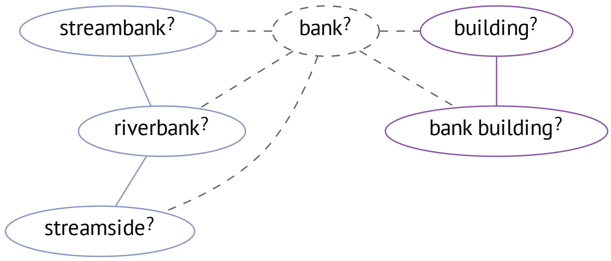


Figure 2
Clustering the neighborhood of the node "bank" of the input graph results in two clusters treated as the non-disambiguated sense contexts: $bank^1 = \{streambank, riverbank, \ldots\}$ and $\{bank^2 = bank\ building, building, \ldots\}$.

Table 2
Example of induced senses for the node "bank" and the corresponding clusters (contexts).

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

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):

$$V_u = \{v \in V : \{u, v\} \in E\} \tag{1}$$

$$E_u = \{\{v, w\} \in E : v, w \in V_u\} \tag{2}$$

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 $u^i \in \text{senses}(u)$, 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):

$$\mathcal{V} = \bigcup_{u \in V} \text{senses}(u) \tag{3}$$

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.

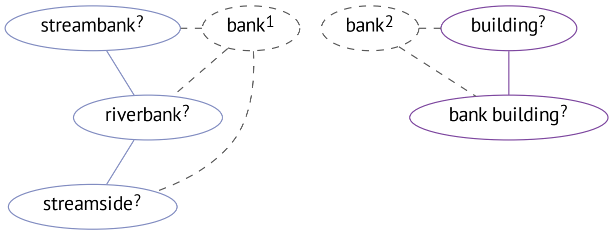


Figure 3
Contexts for two different senses of the node "bank": only its senses *bank*^{1} and *bank*^{2} are currently known, whereas the other nodes in contexts need to be disambiguated.

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 ctx(*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 $\text{sim} : (\text{ctx}(a), \text{ctx}(b)) \to \mathbb{R}$, $\forall \text{ctx}(a), \text{ctx}(b) \subseteq V$.[6] Typical choices for the similarity measure are dot product, cosine similarity, Jaccard index, etc. Hence, we *disambiguate* each context element $v \in \text{ctx}(\hat{u})$:

$$\hat{v} = \arg\max_{v' \in \text{senses}(v)} \text{sim}(\text{ctx}(\hat{u}) \cup \{u\}, \text{ctx}(v')) \tag{4}$$

An example in Figure 4 illustrates the node sense disambiguation process. The context of the sense *bank*^{2} is $\text{ctx}(\textit{bank}^2) = \{\text{building}, \text{bank building}\}$ and the disambiguation target is *building*. Having chosen cosine similarity as the context similarity measure, we compute the similarity between $\text{ctx}(\textit{bank}^2 \cup \{\text{bank}\})$ and the context of every sense of *building* in Table 3: $\cos(\text{ctx}(\textit{bank}^2) \cup \{\text{bank}\}, \text{ctx}(\textit{building}^1)) = \frac{2}{3}$ and $\cos(\text{ctx}(\textit{bank}^2) \cup \{\text{bank}\}, \text{ctx}(\textit{building}^2)) = 0$. 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).

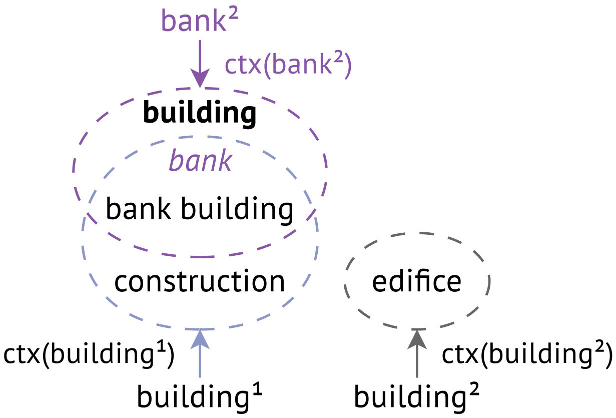


Figure 4
Matching the meaning of the ambiguous node "building" in the context of the sense *bank*^{2}. For matching purposes, the word "bank" is temporarily added into ctx(*bank*^{2}).

context of *bank*^{2}, its first sense, *building*^{1}, should be used because its similarity value is higher.

Finally, we construct a disambiguated context $\widehat{\mathrm{ctx}}(\hat{u}) \subset \mathcal{V}$ that is a sense-aware representation of $\mathrm{ctx}(\hat{u})$. 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 $v \in \mathrm{ctx}(\hat{u})$:

$$\widehat{\mathrm{ctx}}(\hat{u}) = \{\hat{v} \in \mathcal{V} : v \in \mathrm{ctx}(\hat{u})\} \tag{5}$$

As the result of the *local* step, for each node $u \in V$ in the input graph, we induce the $\mathrm{senses}(u) \subset \mathcal{V}$ of nodes and provide each sense $\hat{u} \in \mathcal{V}$ with a disambiguated context $\widehat{\mathrm{ctx}}(\hat{u}) \subseteq \mathcal{V}$.

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):

$$\mathcal{E} = \{\{\hat{u}, \hat{v}\} \in \mathcal{V}^2 : \hat{v} \in \widehat{\mathrm{ctx}}(\hat{u})\} \tag{6}$$

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.

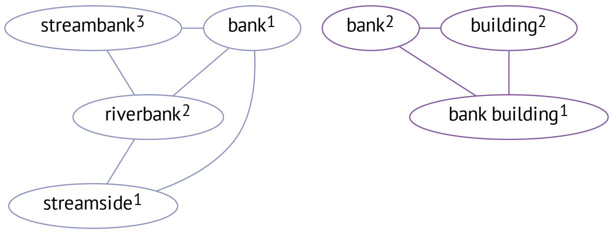


Figure 5
Clustering of the *sense graph* $\mathcal{G}$ yields two clusters, $\{bank^1, streambank^3, riverbank^2, \ldots\}$ and $\{bank^2, bankbuilding^1, building^2, \ldots\}$; if one removes the sense labels, the clusters will overlap, resulting in a *soft* clustering of the input graph G.

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):

$$C = \{\{u \in V : \hat{u} \in \mathcal{C}^i\} \subseteq V : \mathcal{C}^i \in \mathcal{C}\} \tag{7}$$

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

$$\mathcal{E} = \left\{\{u^{\text{senses}[u][v]}, v^{\text{senses}[v][u]}\} \in \mathcal{V}^2 : \{u, v\} \in E\right\} \tag{8}$$

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, $\text{Cluster}_{\text{Local}}$ and $\text{Cluster}_{\text{Global}}$, 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 $O(|V| \deg^2_{\max})$ steps with CW and $O(|V| \deg^3_{\max})$ steps with MCL.

3.5.2 Disambiguation of Neighbors. Let $\text{senses}_{\max}$ be the maximum number of senses for a node and $\text{ctx}_{\max}$ be the maximum size of the node sense context. Thus, this operation takes $O(|V| \times \text{senses}_{\max} \times \text{ctx}_{\max})$ 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 $O(\text{senses}_{\max} \times \text{ctx}_{\max})$ steps per each node in context. Therefore, the whole operation takes $O(|V| \times \text{senses}^2_{\max} \times \text{ctx}^2_{\max})$ steps. Because the maximum number of node senses is observed in a special case when the neighborhood is an unconnected graph, $\text{senses}_{\max} \leq \deg_{\max}$. Given the fact that the maximum context size is observed in a special case when the neighborhood is a fully connected graph, $\text{ctx}_{\max} \leq$

[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.

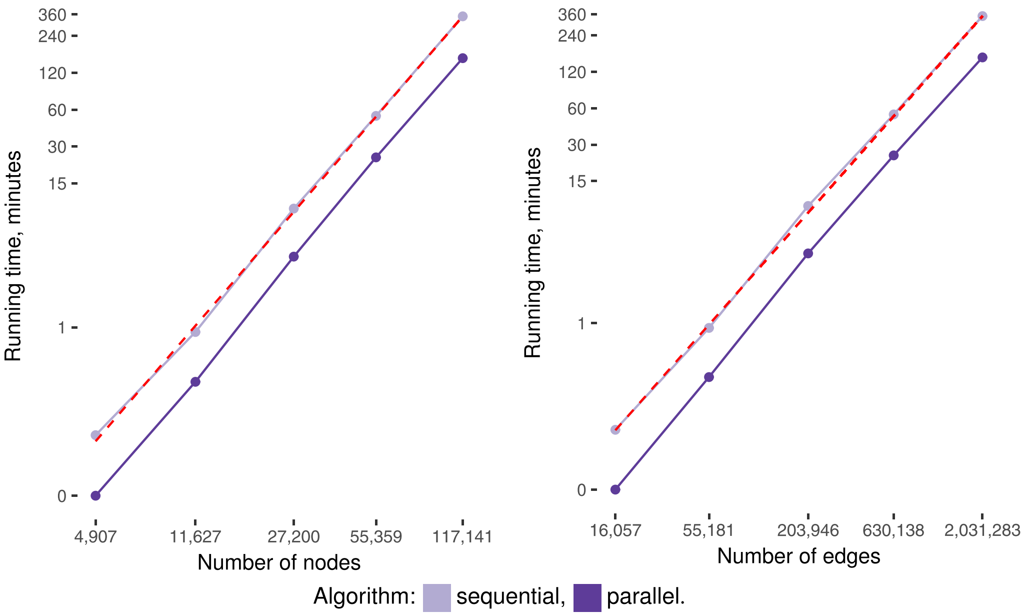


Figure 6
Log-log plots showing growth of the empirical average running time in number of nodes (left) and number of edges (right) of two WATSET[$\mathrm{CW}_{\mathrm{top}}$, $\mathrm{CW}_{\mathrm{top}}$] setups: sequential and parallel. The dashed line is fitted to the running time data of the sequential 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 JWKTL[13] by Zesch, Müller, and Gurevych (2008) and Wikokit[14] 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 *Moneylenders*.[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:

$$\mathrm{TP} = |P \cup P_G| \tag{9}$$

$$\mathrm{FP} = |P \setminus P_G| \tag{10}$$

$$\mathrm{FN} = |P_G \setminus P| \tag{11}$$

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 $\mathrm{Pr} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$, recall as $\mathrm{Re} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$, and F_1-score as $F_1 = \frac{2 \cdot \mathrm{Pr} \cdot \mathrm{Re}}{\mathrm{Pr}+\mathrm{Re}}$. 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 RuWordNet[22] 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 RussNet[23] (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 JWKTL 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 $\text{CPM}_{k=3}$ 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.

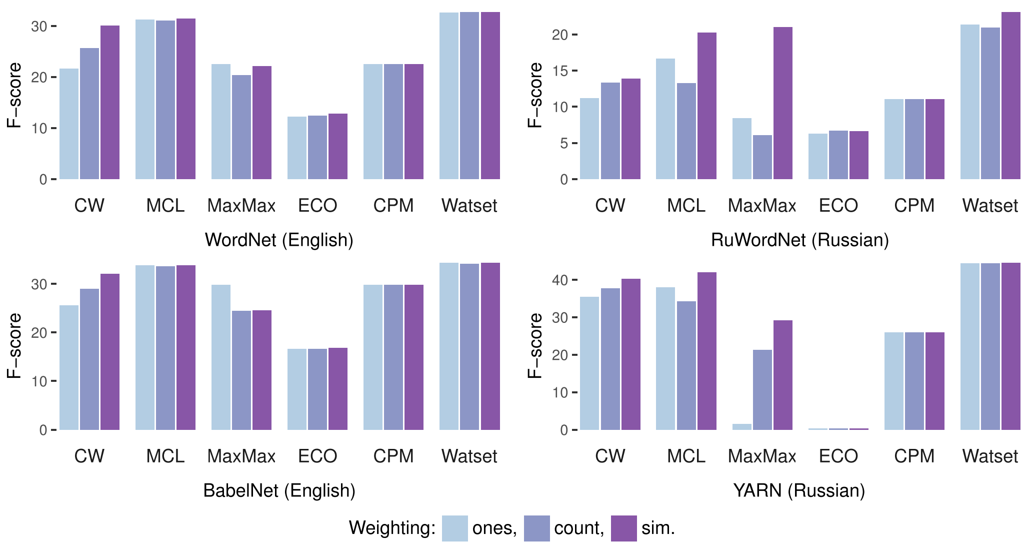


Figure 7
Impact of the different graph-weighting schemas on the performance of synset induction. Each bar corresponds to the top performance of a method in Tables 9 and 10.

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$_{\text{top}}$	243,840	77,879	539,753	28.54	**31.67**	30.02*	32.57	**31.71**	**32.14***
Watset[CW$_{\text{log}}$, MCL]	243,840	164,689	227,906	39.35	27.99	**32.71***	43.94	24.47	31.44*
Watset§[CW$_{\text{top}}$, MCL]	243,840	164,683	227,872	39.17	27.83	**32.54***	43.87	24.40	31.36*
Watset§[CW$_{\text{log}}$, 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$_{\text{lin}}$, MCL]	83,092	58,353	242,615	15.01	**32.55**	**20.55***	46.70	**42.69**	**44.61***
Watset[CW$_{\text{lin}}$, 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$_{\text{lin}}$	83,092	19,124	672,076	8.73	**34.20**	13.91*	36.33	**45.13**	40.25*
Watset§[MCL, CW$_{\text{lin}}$]	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$_{\text{log}}$, 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$_{\text{lin}}$] statistically significantly outperformed all other algorithms, including highly competitive MCL and MaxMax ($p \ll 0.01$). Similarly, on YARN for Russian, Simplified Watset[CW$_{\text{lin}}$, 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, $\cdot$]. 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$_{\text{top/lin/log}}$ 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, incentify, 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.

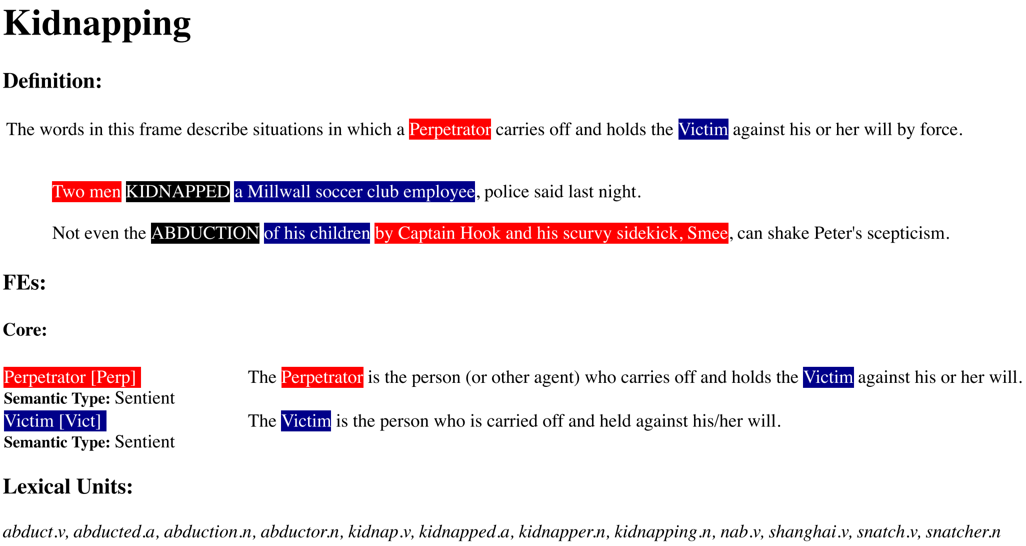
5. Application to Unsupervised Semantic Frame Induction

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] **Semantic Type:** Sentient	The Perpetrator is the person (or other agent) who carries off and holds the Victim against his or her will.
Victim [Vict] **Semantic Type:** Sentient	The Victim is the person who is carried off and held against his/her will.

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 $v \in V \to \vec{v} \in \mathbb{R}^d$ 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 $v \in V \to \vec{v} \in \mathbb{R}^d$,
        the number of nearest neighbors $k \in \mathbb{N}$,
        a graph clustering algorithm Cluster.
Output: a set of triframes $F$.
 1: for all $t = (s, p, o) \in T$ do                          ▷ Embed the triples
 2:     $\vec{t} \leftarrow \vec{s} \oplus \vec{p} \oplus \vec{o}$
 3: $E \leftarrow \{(t, t') \in T^2 : t' \in \mathrm{NN}_k(t), t \neq t'\}$   ▷ Construct edges using nearest neighbors
 4: $G \leftarrow (T, E)$
 5: $F \leftarrow \emptyset$
 6: for all $C^i \in \mathrm{Cluster}(G)$ do                  ▷ Cluster the graph
 7:     $f_s \leftarrow \{s \in V : (s, v, o) \in C^i\}$        ▷ Aggregate subjects
 8:     $f_v \leftarrow \{v \in V : (s, v, o) \in C^i\}$        ▷ Aggregate verbs
 9:     $f_o \leftarrow \{o \in V : (s, v, o) \in C^i\}$        ▷ 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 $\mathrm{NN}_k(t) \subseteq T \setminus \{t\}$. 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 $\vec{t} \in \mathbb{R}^{3d}$

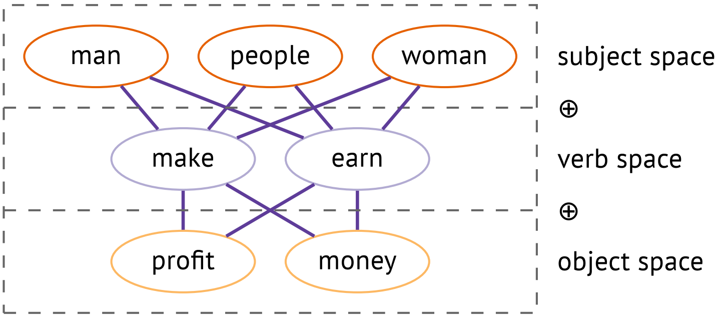


Figure 9
Concatenation of the vectors corresponding to the triple elements, subjects, verbs, and objects, expresses the structural similarity of the triples.

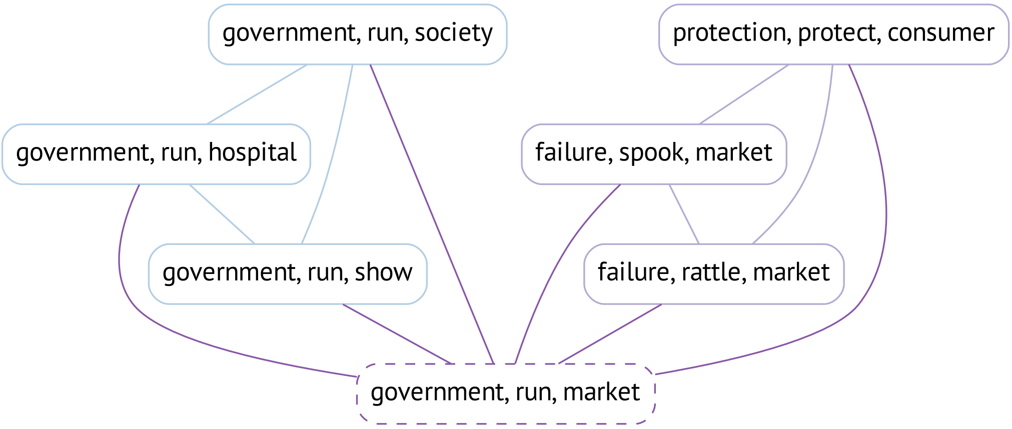


Figure 10
Example of two senses associated with a triple (*government*, *run*, *market*).

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):

$$E = \{(t, t') \in T^2 : t' \in \mathrm{NN}_k(t)\} \tag{12}$$

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 *Singletons* 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 $\delta_{C^i}(C^i \cap C^j_G)$ between each cluster $C^i \in C$ and the gold cluster $C^i_G \in C_G$, which maximizes the overlap with C^i:

$$\mathrm{nmPU} = \frac{1}{|C|} \sum_{i \in \mathbb{N}:|C^i|>1}^{|C|} \max_{1 \leq j \leq |C_G|} \delta_{C^i}(C^i \cap C^j_G) \tag{13}$$

where the weighted overlap is the sum of the weights $C^{i,v}$ for each word $v \in C^i$ in i-th cluster: $\delta_{C^i}(C^i \cap C^j_G) = \sum_{v \in C^i \cap C^j_G} C^{i,v}$. Note that nmPU counts all the singleton clusters as wrong. Similarly, normalized inverse purity (collocation) quantifies the clustering recall:

$$\mathrm{niPU} = \frac{1}{|C_G|} \sum_{j=1}^{|G|} \max_{1 \leq i \leq |C|} \delta_{C^j_G}(C^i \cap C^j_G) \tag{14}$$

Then, nmPU and niPU are combined together as the harmonic mean to yield the overall clustering F_1-score, computed as $\mathrm{F}_1 = 2\frac{\mathrm{nmPU} \cdot \mathrm{niPU}}{\mathrm{nmPU}+\mathrm{niPU}}$, 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 library[31] 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 tricluster 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 ($\text{word}_i\colon \mathit{role}_1, \text{word}_j\colon \mathit{FEE}, \text{word}_k\colon \mathit{role}_2$), where $\text{word}_{i/j/k}$ 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*, kidnap: *FEE*, employee: *Victim*), (men: *Perpetrator*, kidnap: *FEE*, station: *Place*), (employee: *Victim*, kidnap: *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**	**nmPU**	**niPU**	**F_1**	**nmPU**	**niPU**	**F_1**	**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
Singletons	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

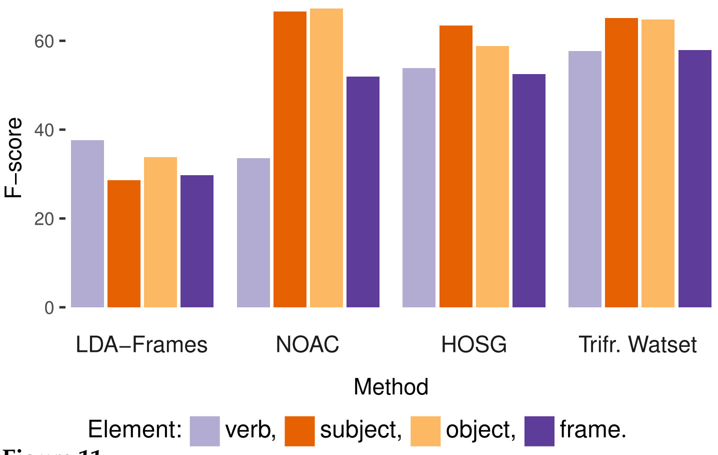


Figure 11
F_1-score values measured on the FrameNet Corpus (Bauer, Fürstenau, and Rambow 2012). Each block corresponds to the top performance of the method in Table 15.

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 *k*-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 `triw2v-watset-n30-top-top-triples.txt` is available in the "Downloads" section of our GitHub repository at `https://github.com/uhh-lt/triframes`.

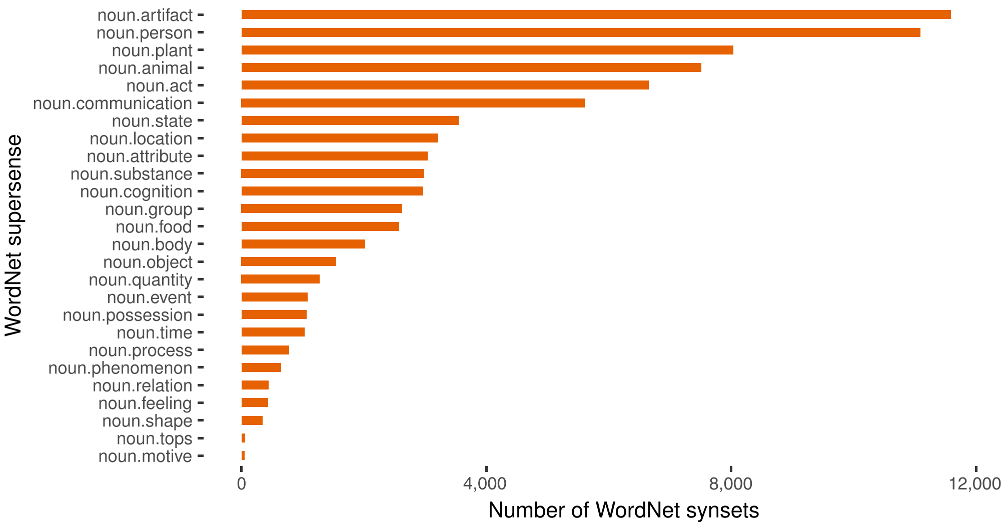


Figure 14
A summary of the noun semantic classes in WordNet supersenses (Ciaramita and Johnson 2003).

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

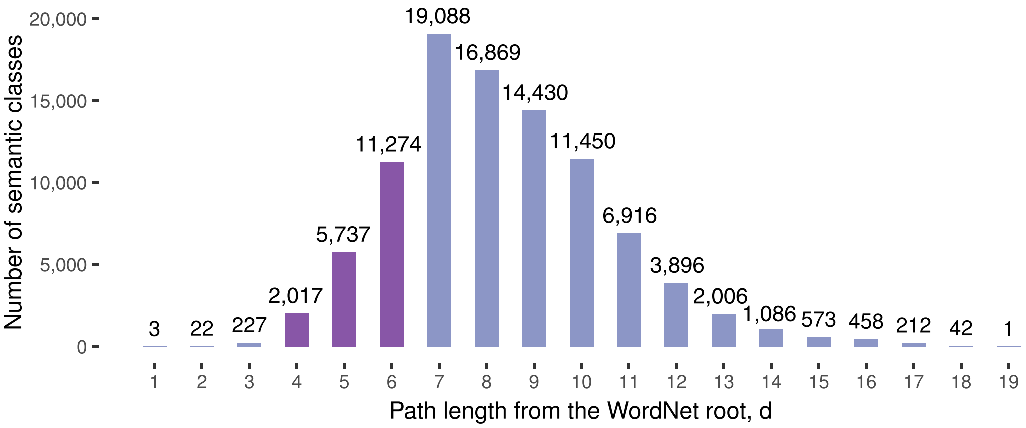


Figure 15
Relationship between the number of semantic classes and path length from the WordNet (Fellbaum 1998) root. We have chosen $d \in \{4, 5, 6\}$ for our experiments.

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 $\{\text{Python}, \text{Perl}\}$. 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 of 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, $\cdot$) or prep_in(code, $\cdot$), 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`.

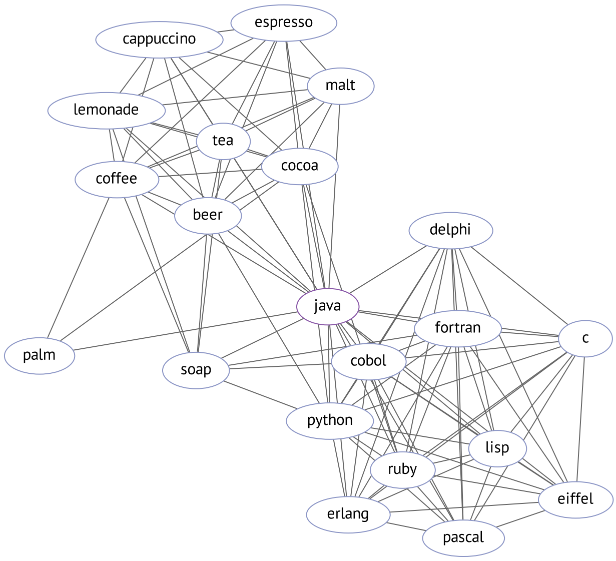


Figure 16
An example of the lexical unit "java" and a part of its neighborhood in a distributional thesaurus. This polysemous word is not disambiguated, so it acts as a hub between two different senses.

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.

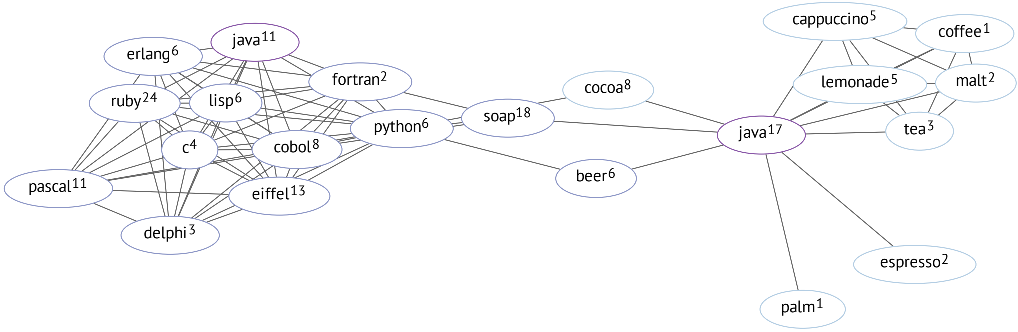


Figure 17
An example of the *sense graph* built by WATSET for two senses of the lexical unit "java" using $\mathrm{CW}_{\log}$ for local clustering. In contrast to Figure 16, in this *disambiguated* distributional thesaurus the node corresponding to the lexical unit "java" is split: $java^{11}$ is connected 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**	$\mathbf{F_1}$	**nmPU**	**niPU**	$\mathbf{F_1}$	**nmPU**	**niPU**	$\mathbf{F_1}$
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 synonymity, 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 monosemeous 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 approach[42] 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 generate*. 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.ldc.upenn.edu/ldc2003t05`.

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, Antonin 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, Roberto Navigli, 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 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 Zvika 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, Srini, 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 Tamas 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 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.

Sarmento, 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, Zeqiu 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 Ponzetto. 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.