Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Alexander Panchenko

INDUCING INTERPRETABLE WORD SENSES FOR WSD AND ENRICHMENT OF LEXICAL RESOURCES

Overview

Overview

- **Inducing word sense representations**:
  - word sense embeddings via retrofitting [Pelevina et al., 2016, Remus and Biemann, 2018];
  - sparse sense representations [Panchenko et al., 2017c];
  - inducing synsets [Ustalov et al., 2017]
  - sense semantic classes [Panchenko et al., 2018]

Overview

- **Inducing word sense representations**:
  - word sense embeddings via retrofitting [Pelevina et al., 2016, Remus and Biemann, 2018];
  - sparse sense representations [Panchenko et al., 2017c];
  - inducing synsets [Ustalov et al., 2017]
  - sense semantic classes [Panchenko et al., 2018]
- **Making induced senses interpretable** [Panchenko et al., 2017b, Panchenko et al., 2017c]

Overview

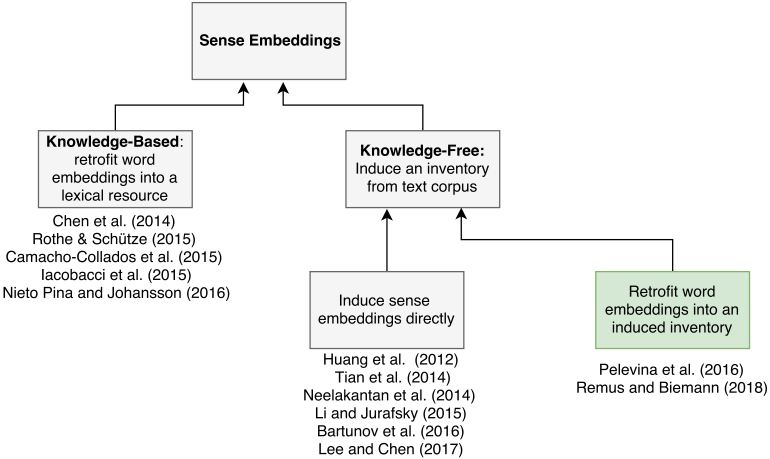
- **Inducing word sense representations**:
  - word sense embeddings via retrofitting [Pelevina et al., 2016, Remus and Biemann, 2018];
  - sparse sense representations [Panchenko et al., 2017c];
  - inducing synsets [Ustalov et al., 2017]
  - sense semantic classes [Panchenko et al., 2018]
- **Making induced senses interpretable** [Panchenko et al., 2017b, Panchenko et al., 2017c]
- **Linking induced word senses to lexical resources** [Faralli et al., 2016, Panchenko et al., 2017a, Biemann et al., 2018]

Inducing word sense representations

Word vs sense embeddings

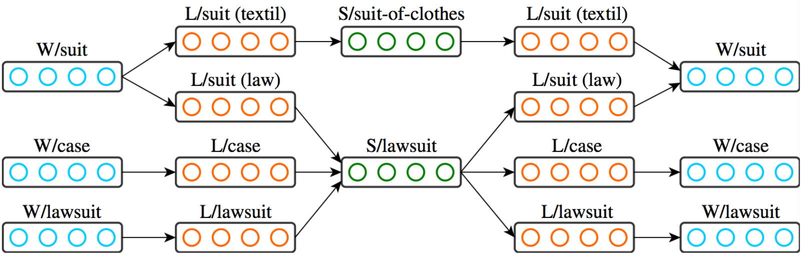
Word vs sense embeddings

Related work



Related work: knowledge-based

- **AutoExtend** [Rothe and Schütze, 2015]



* image is reproduced from the original paper

Related work: knowledge-free

- **Adagram** [Bartunov et al., 2016]
- Multiple vector representations θ for each word:

Related work: knowledge-free

- **Adagram** [Bartunov et al., 2016]
- Multiple vector representations θ for each word:

$$p(Y, Z, \beta | X, \alpha, \theta) = \prod_{w=1}^{V} \prod_{k=1}^{\infty} p(\beta_{wk} | \alpha) \prod_{i=1}^{N} [p(z_i | x_i, \beta) \prod_{j=1}^{C} p(y_{ij} | z_i, x_i, \theta)],$$

  - z_i – a hidden variable: a sense index of word x_i in context C;
  - α – a meta-parameter controlling number of senses.

Related work: knowledge-free

- **Adagram** [Bartunov et al., 2016]
- Multiple vector representations θ for each word:

$$p(Y, Z, \beta | X, \alpha, \theta) = \prod_{w=1}^{V} \prod_{k=1}^{\infty} p(\beta_{wk} | \alpha) \prod_{i=1}^{N} [p(z_i | x_i, \beta) \prod_{j=1}^{C} p(y_{ij} | z_i, x_i, \theta)],$$

  - z_i – a hidden variable: a sense index of word x_i in context C;
  - α – a meta-parameter controlling number of senses.
- **See also**: [Neelakantan et al., 2014] and [Li and Jurafsky, 2015]

Related work: word sense induction

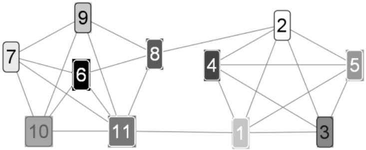
- Word sense induction (WSI) based on **graph clustering**:
  - [Lin, 1998]
  - [Pantel and Lin, 2002]
  - [Widdows and Dorow, 2002]
  - **Chinese Whispers [Biemann, 2006]**
  - [Hope and Keller, 2013]

Related work: Chinese Whispers#1

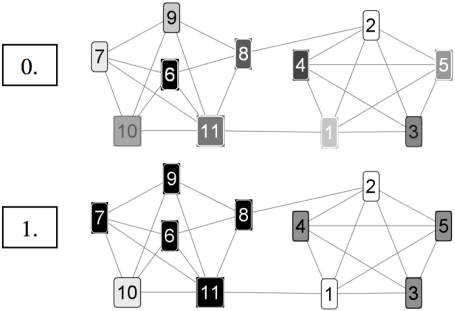
* source of the image: `http://ic.pics.livejournal.com/blagin_anton/33716210/2701748/2701748_800.jpg`

Related work: Chinese Whispers#2

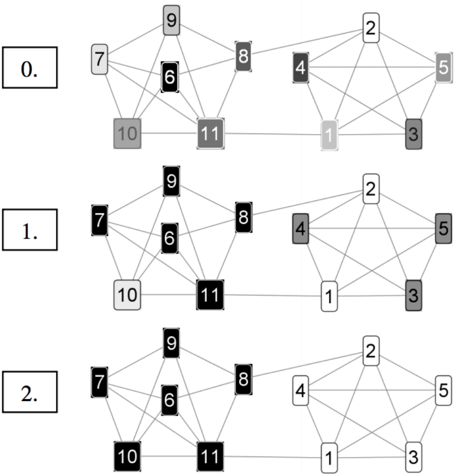
0.



Related work: Chinese Whispers#2



Related work: Chinese Whispers#2



Sense embeddings using retrofitting

RepL4NLP@ACL'16 [Pelevina et al., 2016], **LREC'18** [Remus and Biemann, 2018]

Prior methods:

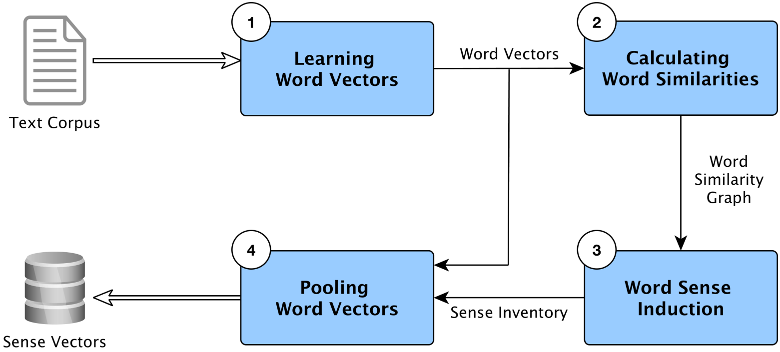
- Induce inventory by clustering of word instances
- Use existing sense inventories

Our method:

- **Input:** word embeddings
- **Output:** word sense embeddings
- **Word sense induction** by clustering of word ego-networks

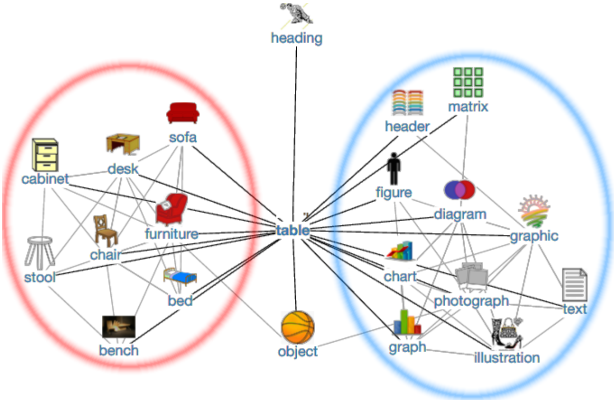
Sense embeddings using retrofitting

- From word embeddings to sense embeddings



Sense embeddings using retrofitting

- Word sense induction using ego-network clustering



Sense embeddings using retrofitting

- Neighbours of Word and Sense Vectors

Vector	Nearest Neighbors
table	tray, bottom, diagram, bucket, brackets, stack, basket, list, parenthesis, cup, saucer, pile, playfield, bracket, pot, drop-down, cue, plate

Sense embeddings using retrofitting

- Neighbours of Word and Sense Vectors

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

Sense embeddings using retrofitting

Word Sense Disambiguation

1. **Context extraction**: use context words around the target word
2. **Context filtering**: based on context word's relevance for disambiguation
3. **Sense choice in context**: maximise similarity between a context vector and a sense vector

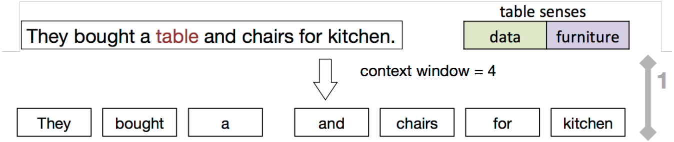
Sense embeddings using retrofitting

They bought a table and chairs for kitchen.

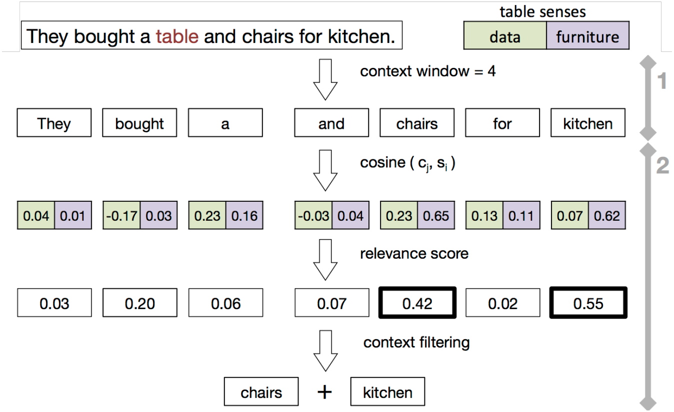
table senses

data	furniture

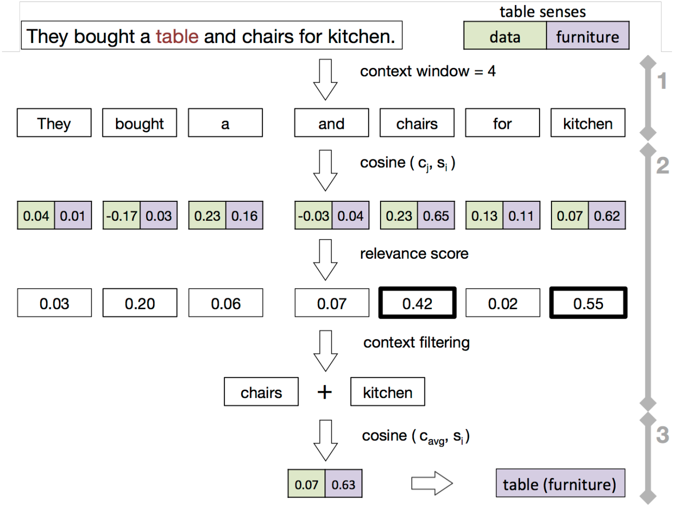
Sense embeddings using retrofitting



Sense embeddings using retrofitting



Sense embeddings using retrofitting



Sense embeddings using retrofitting

Evaluation on SemEval 2013 Task 13 WSI&D:

Model	Jacc.	Tau	WNDCG	F.NMI	F.B-Cubed
AI-KU (add1000)	0.176	0.609	0.205	0.033	0.317
AI-KU	0.176	0.619	0.393	0.066	0.382
AI-KU (remove5-add1000)	0.228	0.654	0.330	0.040	0.463
Unimelb (5p)	0.198	0.623	0.374	0.056	0.475
Unimelb (50k)	0.198	0.633	0.384	0.060	0.494
UoS (#WN senses)	0.171	0.600	0.298	0.046	0.186
UoS (top-3)	0.220	0.637	0.370	0.044	0.451
La Sapienza (1)	0.131	0.544	0.332	–	–
La Sapienza (2)	0.131	0.535	0.394	–	–
AdaGram, α = 0.05, 100 dim	0.274	0.644	0.318	0.058	0.470
w2v	0.197	0.615	0.291	0.011	0.615
w2v (nouns)	0.179	0.626	0.304	0.011	0.623
JBT	0.205	0.624	0.291	0.017	0.598
JBT (nouns)	0.198	0.643	0.310	0.031	0.595
TWSI (nouns)	0.215	0.651	0.318	0.030	0.573

Sense embeddings using retrofitting

LREC'2018 [Remus and Biemann, 2018]

	AUTOEXTEND	ADAGRAM	SGNS	.	GLOVE	.	SYMPAT	.	LSABOW	.	LSAHAL	.	PARAGRAMSL	.
SIMLEX999	0.45	0.29	0.44		0.37		0.54		0.30		0.27		0.68	
MEN	0.72	0.67	0.77		0.73		0.53		0.67		0.71		0.77	
SIMVERB	0.43	0.27	0.36		0.23		0.37		0.15		0.19		0.53	
WORDSIM353	0.58	0.61	0.70		0.61		0.47		0.67		0.59		0.72	
SIMLEX999-N	0.44	0.33	0.45		0.39		0.48		0.32		0.34		0.68	
MEN-N	0.72	0.68	0.77	____	0.76	____	0.57	____	0.71	____	0.73	____	0.78	____

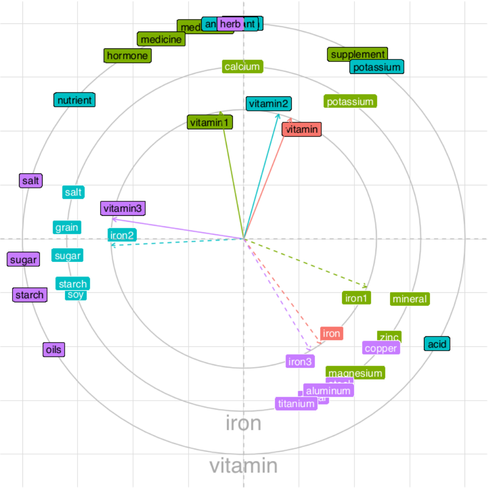
Sense embeddings using retrofitting

LREC'2018 [Remus and Biemann, 2018]

	AUTOEXTEND	ADAGRAM	SGNS	SGNS+SENSES	GLOVE	GLOVE+SENSES	SYMPAT	SYMPAT+SENSES	LSABOW	LSABOW+SENSES	LSAHAL	LSAHAL+SENSES	PARAGRAMSL	PARAGRAMSL+SENSES
SIMLEX999	0.45	0.29	0.44	**0.46**	0.37	**0.41**	0.54	**0.55**	0.30	**0.39**	0.27	**0.38**	**0.68**	0.64
MEN	0.72	0.67	0.77	**0.78**	0.73	**0.77**	0.53	**0.68**	0.67	**0.70**	0.71	**0.74**	0.77	**0.80**
SIMVERB	0.43	0.27	0.36	**0.39**	0.23	**0.30**	0.37	**0.45**	0.15	**0.22**	0.19	**0.28**	0.53	0.53
WORDSIM353	0.58	0.61	**0.70**	0.69	0.61	**0.65**	0.47	**0.62**	**0.67**	0.66	0.59	**0.63**	0.72	**0.73**
SIMLEX999-N	0.44	0.33	0.45	**0.50**	0.39	**0.47**	0.48	**0.55**	0.32	**0.46**	0.34	**0.44**	**0.68**	0.66
MEN-N	0.72	0.68	0.77	**0.79**	0.76	**0.80**	0.57	**0.74**	0.71	**0.73**	0.73	**0.76**	0.78	**0.81**

- Sense-aware similarities are marked with **+SENSES**.
- These results are using a sense inventory based on **sparse dependency features** (JoBimText).

Sense embeddings using retrofitting



- Word and sense embeddings of words **iron** and **vitamin**.

LREC'18 [Remus and Biemann, 2018]

Sparse sense representations

Sparse sense representations

Watset: synset induction

Watset: synset induction

Watset: synset induction

Induction of sense semantic classes

Induction of sense semantic classes

Induction of sense semantic classes

Making induced senses interpretable

Making induced senses interpretable

Knowledge-based sense representations are **interpretable**

• bn:01713224n • NOUN • Named Entity • Categories: High-level programming languages, Dutch inventions, Class-based programming languages, Cros platform free software...

EN **Python (programming language)** · **/usr/bin/python** · **/usr/local/bin/python** · **Python language** · **Python programming language**

Python is a widely used general-purpose, high-level programming language. *Wikipedia*

More definitions

IS A	programming language • free software • scripting language
HAS PART	pandas
HAS KIND	Stackless Python
DESIGNER	Guido van Rossum
DEVELOPER	Python Software Foundation • Guido van Rossum
DIALECTS	Cython • Stackless Python
INFLUENCED BY	ALGOL 68 • alphabet • ruby
LICENSE	Python Software Foundation License

More relations

EXPLORE NETWORK

Making induced senses interpretable

Knowledge-free sense representations are **uninterpretable**

```
In [11]: sv.syn0[sv.vocab["python#2"].index]
Out[11]:
array([-0.0493343 , -0.02244579,  0.02296794,  0.03484775,  0.0404554 ,
        0.04304857, -0.02211852, -0.02118347, -0.03212074, -0.01202453,
        0.01206081,  0.05609602, -0.05950832,  0.00859888, -0.01051112,
        0.03177784, -0.06489294,  0.03833736,  0.05437034, -0.01451268,
       -0.02419239, -0.03195219,  0.0620546 ,  0.10284331,  0.07430374,
       -0.04109243, -0.0118133 ,  0.05401124,  0.05283536,  0.00873093,
        0.03662092,  0.03762468,  0.02368712, -0.03980339,  0.02791001,
        0.02529952, -0.02255581, -0.00925604, -0.03940469, -0.02855149,
                                              ...
       -0.08179335,  0.02319797, -0.0167018 ,  0.04818865, -0.06946786,
       -0.06530198,  0.00522405, -0.0336296 , -0.05401101,  0.01190361], dtype=float32)
```

Making induced senses interpretable

Making induced senses interpretable

Making induced senses interpretable

Linking induced senses to resources

Linking induced senses to resources

Linking induced senses to resources

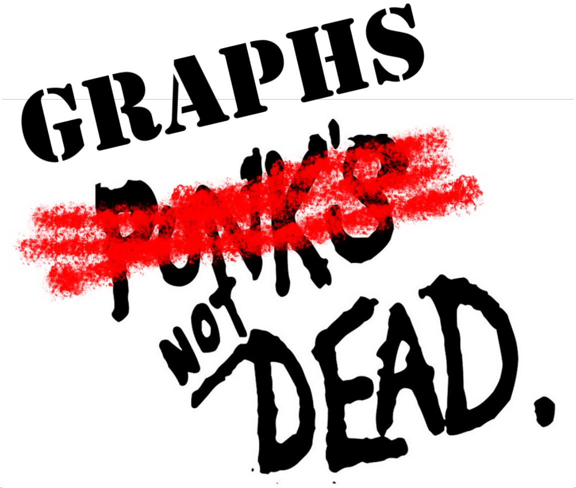
Linking induced senses to resources

Linking induced senses to resources

Linking induced senses to resources

Conclusion

Vectors + Graphs = ♡



Take home messages

- We can **induce word senses**, **synsets** and **semantic classes** in a knowledge-free way using **graph clustering** and **distributional models**.

Take home messages

- We can **induce word senses**, **synsets** and **semantic classes** in a knowledge-free way using **graph clustering** and **distributional models**.
- We can make the **induced word senses interpretable** in a knowledge-free way with **hypernyms**, **images**, **definitions**.

Take home messages

- We can **induce word senses**, **synsets** and **semantic classes** in a knowledge-free way using **graph clustering** and **distributional models**.
- We can make the **induced word senses interpretable** in a knowledge-free way with **hypernyms**, **images**, **definitions**.
- We can **link induced senses to lexical resources** to
  - improve **performance of WSD**;
  - **enrich lexical resources** with emerging senses.

An ongoing shared task on WSI&D

- Participate in an **ACL SIGSLAV** sponsored shared task on **word sense induction and disambiguation** for Russian!
- **More details**: `http://russe.nlpub.org/2018/wsi`

Acknowledgments

Thank you! Questions?

This research was supported by

DFG Deutsche Forschungsgemeinschaft

DAAD Deutscher Akademischer Austausch Dienst German Academic Exchange Service

Bartunov, S., Kondrashkin, D., Osokin, A., and Vetrov, D. (2016).
Breaking sticks and ambiguities with adaptive skip-gram.
In *Artificial Intelligence and Statistics*, pages 130–138.

Biemann, C., Faralli, S., Panchenko, A., and Ponzetto, S. P. (2018).
A framework for enriching lexical semantic resources with distributional semantics.
In *Journal of Natural Language Engineering*, pages 56–64. Cambridge Press.

Faralli, S., Panchenko, A., Biemann, C., and Ponzetto, S. P. (2016).
Linked disambiguated distributional semantic networks.
In *International Semantic Web Conference*, pages 56–64. Springer.

Panchenko, A., Faralli, S., Ponzetto, S. P., and Biemann, C. (2017a).

Using linked disambiguated distributional networks for word sense disambiguation.
In *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications*, pages 72–78, Valencia, Spain. Association for Computational Linguistics.

Panchenko, A., Marten, F., Ruppert, E., Faralli, S., Ustalov, D., Ponzetto, S. P., and Biemann, C. (2017b).
Unsupervised, knowledge-free, and interpretable word sense disambiguation.
In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 91–96, Copenhagen, Denmark. Association for Computational Linguistics.

Panchenko, A., Ruppert, E., Faralli, S., Ponzetto, S. P., and Biemann, C. (2017c).
Unsupervised does not mean uninterpretable: The case for word sense induction and disambiguation.
In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1,*

Long Papers, pages 86–98, Valencia, Spain. Association for Computational Linguistics.

Panchenko, A., Ustalov, D., Faralli, S., Ponzetto, S. P., and Biemann, C. (2018).
Improving hypernymy extraction with distributional semantic classes.
In *Proceedings of the LREC 2018*, Miyazaki, Japan. European Language Resources Association.

Pelevina, M., Arefiev, N., Biemann, C., and Panchenko, A. (2016).
Making sense of word embeddings.
In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 174–183, Berlin, Germany. Association for Computational Linguistics.

Remus, S. and Biemann, C. (2018).
Retrofittingword representations for unsupervised sense aware word similarities.

In *Proceedings of the LREC 2018*, Miyazaki, Japan. European Language Resources Association.

Rothe, S. and Schütze, H. (2015).
Autoextend: Extending word embeddings to embeddings for synsets and lexemes.
In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1793–1803, Beijing, China. Association for Computational Linguistics.

Ustalov, D., Panchenko, A., and Biemann, C. (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, Canada. Association for Computational Linguistics.