Vectors or Graphs?
On Differences of Representations for Distributional Semantic Models

Chris Biemann
Language Technology Group
Computer Science Dept.
University of Hamburg, Germany
biemann@uni-hamburg.de

Abstract

Distributional Semantic Models (DSMs) have recently received increased attention, together with the rise of neural architectures for scalable training of dense vector embeddings. While some of the literature even includes terms like 'vectors' and 'dimensionality' in the definition of DSMs, there are some good reasons why we should consider alternative formulations of distributional models. As an instance, I present a scalable graph-based solution to distributional semantics. The model belongs to the family of 'count-based' DSMs, keeps its representation sparse and explicit, and thus fully interpretable. I will highlight some important differences between sparse graph-based and dense vector approaches to DSMs: while dense vector-based models are computationally easier to handle and provide a nice uniform representation that can be compared and combined in many ways, they lack interpretability, provenance and robustness. On the other hand, graph-based sparse models have a more straightforward interpretation, handle sense distinctions more naturally and can straightforwardly be linked to knowledge bases, while lacking the ability to compare arbitrary lexical units and a compositionality operation. Since both representations have their merits, I opt for exploring their combination in the outlook.

1 Introduction

Rooted in Structural Linguistics (de Saussure, 1966; Harris, 1951), *Distributional Semantic Models* (DSMs, see e.g. (Baroni and Lenci, 2010)) characterize the meaning of lexical units by the contexts they appear in, cf. (Wittgenstein, 1963; Firth, 1957). Using the duality of form and contexts, forms can be compared along their contexts (Miller and Charles, 1991), giving rise to the field of Statistical Semantics. A data-driven, unsupervised approach to representing word meaning is attractive as there is no need for laborious creation of lexical resources. Further, these approaches naturally adapt to the domain or even language at hand. Desirable, in general, is a model that provides a firm basis for a wider range of (semantic) tasks, as opposed to specialised solutions on a per-task basis.

While most approaches to distributional semantics rely on dense vector representations, the reasons for this seem rather technical than well-justified. To de-bias the discussion, I propose a competitive graph-based formulation. Since all representations have advantages and disadvantages, I will discuss some ways of how to fruitfully combine graphs and vectors in the future.

1.1 Vectors – a solution to Plato's Problem?

Vector space models have a long tradition in Information Retrieval (Salton et al., 1975), and heavily influence the way we think about representing documents and terms today. The core idea is to represent each document with a bag-of-words vector of $|V|$ dimensions with vocabulary V, counting how often

Proceedings of the Workshop on Cognitive Aspects of the Lexicon,
pages 1–7, Osaka, Japan, December 11-17 2016.

each word appears in the respective document. Queries, which are in fact very short documents, can be matched to documents by appropriately comparing their vectors. Since V is large, the representation is sparse – most entries in the vectors are zero. Note, however, that zeros are not stored in today's indexing approaches. When Deerwester et al. (1990) introduced Latent Semantic Analysis (LSA), its major feature was to reduce the dimensionality of vectors, utilising the entirety of all documents for characterising words by the documents they appear in, and vice versa. Dimensionality reduction approaches like Singular Value Decomposition (SVD) and Principal Component Analysis (PCA) map distributionally similar words (occurring in similar contexts) to similar lower-dimensional vectors, where the dimensionality typically ranges from 200 to 10'000. Such a representation is dense: there are virtually no zero entries in these vectors. A range of more recent models, such as Latent Dirichlet Allocation (LDA), are characterised in the same way – variants are distinguished by the notion of context (document vs. window-based vs. structured by grammatical dependencies) and the mechanism for dimensionality reduction. With the advent of neural embeddings such as word2vec (Mikolov et al., 2013), a series of works showed modest but significant advances in semantic tasks over previous approaches. Levy and Goldberg (2014b), however, showed that there is no substantial representational advance in neural embeddings, as they approximate matrix factorisation, as used in LSA. The advantage of word2vec is rather its efficient and scalable implementation that enables the processing of larger text collections. Improvements on task performance can mostly be attributed to better tuning of hyperparameters[1] – which however overfits the DSM to a task at hand, and defies the premise of unsupervised systems of not needing (hyper)supervision.

But there is a problem with all of these approaches: the *fallacy of dimensionality*[2], following from a simplification that we should not apply without being aware of its consequences: there is no 'appropriate number' of dimensions in natural language, because natural language follows a scale-free distribution on all levels (e.g. (Zipf, 1949; Steyvers and Tenenbaum, 2005; Mukherjee et al., 2008), inter al.). Thus, a representation with a fixed number of dimensions introduces a granularity – 'major' dimensions encode the most important distinctions while 'minor' distinctions in the data cannot be modelled if the granularity is too coarse. This is why the recommended number of dimensions depends on the task, the dataset's size and even the domain. In principle, there are two conclusions from studies that vary the number of dimensions to optimise some sort of a score: (a) in one type of study, there is a sweet spot in the number of dimensions, typically between 50 and 2000. This means that the dimension is indeed task-dependent, (b) the 'optimal' number of dimensions is the highest number tested, indicating that it probably would have been better to keep a sparse representation. Interestingly, the most frequent reason researchers state, if asked why they did not use a sparse representation, is a technical one: many machine learning and statistical libraries do not natively operate on sparse representations, thus run out of memory when trying to represent all those zeros.

2 Graph-based Sparse Representations

Since I am proposing to de-bias the discussion on DSMs from the domination of vectors towards a more balanced view, I am exemplifying a graph-based DSM in this section. The JoBimText (Biemann and Riedl, 2013) framework is a scalable graph-based DSM implementation, developed in cooperation with IBM Research (Gliozzo et al., 2013). It is defined rather straightforwardly: lexical items $j \in J$ are represented by their p most salient contexts B_j, where saliency is measured by frequency or a statistical measure that prefers frequent co-occurrence, such as LMI (Evert, 2004) or LL (Dunning, 1993). Similarity of lexical items is defined as the overlap count of their respective contexts: $sim(j_k, j_l) = |(x|x \in B_{j_k} \& x \in B_{j_l})|$. We call the graph of all lexical items with edges weighted by this similarity a distributional thesaurus (DT). Despite its simplicity, or maybe because of that, this DSM compares favourably to other DSMs (Riedl, 2016), including Lin's thesaurus (Lin, 1998), Curran's measure (Curran, 2004), and word embeddings (Mikolov et al., 2013; Levy and Goldberg, 2014a) on word similarity tasks, especially for large data. It was further successfully used for word expansion in word sense disambiguation (Miller et al., 2012), as a

[1] "If you want to get good results, you should tune your hyperparameters. And if you want to make good science, don't forget to tune your baselines' hyperparameters too!" - Omer Levy, pers. communication

[2] not to be confused with the curse of dimensionality, which refers to adverse phenomena when representing problems in too high-dimensional spaces

entry	similar terms	hypernyms	context
mouse:NN:0	rat:NN**:0**, rodent:NN**:0**, monkey:NN**:0**, ...	animal:NN**:0**, species:NN**:1**, ...	rat::NN:conj_and, white-footed:JJ:amod, ...
mouse:NN:1	keyboard:NN**:1**, computer:NN**:0**, printer:NN**:0** ...	device:NN**:1**, equipment:NN**:3**, ...	click:NN:-prep_of, click:NN:-nn, ...
keyboard:NN:0	piano:NN**:1**, synthesizer:NN**:2**, organ:NN**:0** ...	instrument:NN**:2**, device:NN**:3**, ...	play:VB:-dobj, electric:JJ:amod, ..
keyboard:NN:1	keypad:NN**:0**, mouse:NN**:1**, screen:NN**:1** ...	device:NN**:1**, technology:NN**:0** ...	computer:NN:nn, qwerty:JJ:amod ...

Table 1: Examples of PCZ entries for "mouse:NN" and "keyboard:NN" based on dependency contexts (cf. (Erk and Padó, 2008)) from a newspaper corpus. Trailing numbers indicate sense identifiers. Similarity and context scores are not shown for brevity.

feature for lexical substitution (Szarvas et al., 2013), for multiword identification (Riedl and Biemann, 2015), decompounding (Riedl and Biemann, 2016) and for resolving bridging mentions in co-reference (Feuerbach et al., 2015).

The key to a scalable implementation is rooted in the pruning parameter p (typically p=1000), which has two functions: it reduces noise in the representations by only keeping the most salient contexts, and it limits the size of the representation, which is a list of key-value-pairs of fixed length (as opposed to a vector of fixed length). In other words: While there is a maximum size of the representation, as given by p, it is not the case that the information is compressed in a vector of fixed dimensionality, since 'dimensions', if one wants to call them such, are different for each represented item.

Of course, it would be possible to represent item-contexts or the distributional thesaurus in sparse matrices of very high dimensionality, but this view would not take the inherent sparseness into account and might obscure possible optimizations.

Using the JoBimText DSM as a core, we extend this model in several ways. First, we perform word sense induction (WSI) on the ego-networks of lexical items in the DT (Biemann, 2006), utilising the property of many graph clustering algorithms that do not require the number of clusters as input (like e.g. k-Means). Further, we add taxonomic links (hypernyms) from Hearst-pattern-like extractions (Hearst, 1992). WSI allows us to disambiguate the model, which results in what we call a Proto-conceptualization (PCZ) (Faralli et al., 2016), see Table 1. The PCZ consists of entries that correspond to word senses, a list of similar senses, a list of hypernyms and a list of contexts that are salient for the sense. Note that it is straightforward to add these and other typed, weighted relationships in a graph-based framework, cf. (Hovy, 2010). Furthermore, it is straightforward to link this kind of structure to existing structured resources, such as lexical-semantic networks and ontologies, see (Pavel and Euzenat, 2013; Faralli et al., 2016).

While it is possible to represent the similarity graph of terms or concepts of a graph-based DSM with real-valued matrices, it is not straightforward to convert it into a metric space since the overlap similarity measure is not a distance measure. For example[3], we find the most similar word to "anaconda" to be "python" in the snake sense with a similarity score of 36 and "snake" with a similarity score of 31. However, "python" snake's list of most similar terms starts with "snake, serpent, rattlesnake, cobra...", with "anaconda" appearing at rank 26.

3 Comparison of Graph-based and Vector-based DSMs

Above, I already hinted at fundamental differences between vector-based (VDSMs) and graph-based (GDSMs) distributional semantic models. In this section, these differences and their consequences are described in more detail. Most differences are rooted in the fact that VDSMs encode lexical items in a metric space, where a point in the n-dimensional space corresponds to the coordinates given by the n-dimensional vector. This is not the case for GDSMs for lack of fixed dimensionality. In all of the aspects discussed below, there exist solutions for both representations, but in many cases, one of the representations is more suitable than the other.

Word Similarity In word relatedness or similarity evaluations, where the global similarity ranking of word pairs should be predicted by the DSM, VDSMs excel since the graph-based model does not relate lexical items that are dissimilar at all, therefore not being able to discern a difference in degree of

[3]using the Google Books Syntactic Dependencies model on www.jobimtext.org/jobimviz-web-demo/

relatedness e.g. between rooster:voyage and asylum:fruit (from RG65; (Rubenstein and Goodenough, 1965)). On the other hand, the ability of the GDSM only return a set of minimally similar items has been experimentally shown to be advantageous when using DSM similarity for lexical expansion (cf. (Miller et al., 2012)).

Similarity Computation and Semantic Neighbourhood Similarity computation in the metric space of the VDSM is computationally expensive and needs engineering solutions like K-d-trees (Bentley, 1990) or approximation (Sugawara et al., 2016) to make it feasible to return the top-n-similar list of items, which is a frequently used function in statistical semantics. In GDSMs, on the other hand, similarity is directly read off the representation. Pre-computation of all similarities in VDSMs is possible, but does not scale well, cf. (Panchenko et al., 2016).

Word Sense Representations Another consequence of the metric space is that neighbourhoods of lexical items are populated with similar lexical items across all frequency bands. This leads to the following situation when trying to induce word senses: Suppose we hypothesise for a lexical item like "bank" that it has more than one sense and we want to cluster the neighbourhood to get two sense representations. As for most ambiguous words, the sense distribution is biased: in our hypothetical collection, the monetary sense of bank is much better represented than the river bank sense. In this situation, the vector for "bank" is surrounded by other money-bank-terms (such as names of banks). The larger the underlying corpus, the higher is the amount of these terms, most of them rare (see e.g. (Pelevina et al., 2016)). We either do not find river-bank terms in the neighbourhood or we have to extend the neighbourhood until we pull in a lot of unrelated words into our subspace we use for clustering. This might be a reason why in word sense induction, graph-based algorithms are very popular while there are only few approaches that determine the number of sense embeddings per item automatically (as e.g. (Neelakantan et al., 2014)).

Word analogy and other arithmetics Word analogy tasks are a classic use-case for word embeddings, and there are further works, which learn vector operations that represent semantic relations. While many of these approaches in fact learn prototypical heads of the respective relation (Levy et al., 2015), word analogy and relational arithmetics are much less straightforward in GDSMs.

Compositionality This is another task where the VDSM representation is more suited than GDSMs. While in general, a scalable computation in GDSMs allows to compute representation and similarities for frequent multi-word units (Riedl and Biemann, 2015), the computation of compositional vectors from single vectors in VDSMs is more attractive since it generalises to unseen combinations, even phrases and sentences (Bentivogli et al., 2016).

Interpretability and Robustness of Representation The lack of interpretability of vectors and their dimensions is one of the strongest points of critique on dense vectors: while sometimes, post-hoc explanations for some of the dimensions are found, it holds in general that most or nearly all latent dimensions have no direct interpretation, and running the same model on a somewhat different collection would yield entirely different dimensions and embeddings. This is where sparse models shine, as their representations are readable. For example, it is possible to query the GDSM described above *why* anaconda and python are similar (because they coil up, are snakes, swallow, digest, gorge, tighten, and co-occur in conjunctions with other snakes, easily readable off the shared context representation) – and the same 'reasons' for similarity will be found in other corpora as well, assuming they contain a sufficient amount of snakes.

Learnability and Cognitive Plausibility A cognitively plausible model should be able to learn continuously and iteratively from an input stream of language. This point is not well-addressed by both representations. While it is agreed upon that human brains operate on distributed neural representations, this is where the commonalities between humans and static, per-task neural architectures already end. One major divergence lies in the epochal training, which humans do not need, especially when extending their vocabulary. Dense vector representations are either produced by a single operation that requires the entire

corpus and vocabulary to be known beforehand (e.g. LSA) or by sampling methods that are obtained by several iterations over the input data, which also require a fixed vocabulary. Count-based sparse methods do not suffer from the fixed vocabulary restriction; however, it is also implausible that the sparse full co-occurrence counts are stored, and most of the current implementations are technically implemented in batch mode, not providing the possibility to update the model through processing further material.

For restrictions of space, this list of differences ends here. There are different criteria and use-cases for DSMs, and there are solutions or at least circumventions for most of the critical points I have risen. However, what should become clear is, that there is a substantial difference and some representations are in fact more adequate than others, depending on the scenario or task.

4 Conclusions and Outlook

Where do we go from here? If this position paper has convinced you as a reader to re-visit the assumption that DSMs *must* be represented in vector spaces, then I have already reached my goal. Now that we hopefully agree that there is value in both vector-based and graph-based representations, the next natural question is how to combine them to get the best of both worlds. Ideally, depending on the tsk, problem, and engineering constraints, it would be desirable to switch between both representations, or to inform one another at construction time.

A starting point to a combination might be to break down methods that use DSMs into their parts and to gauge which representation is more suitable. For example, take word sense induction and disambiguation: As mentioned above, it might be advantageous to cluster graphs instead of vectors because there are straightforward methods that do not require the number of clusters to be set beforehand (which is a 'big no-no' in WSI) and because vector space neighbourhoods of words with biased sense distributions might be overpopulated by the dominant sense. However, for disambiguation, it might be an advantage to use dense representations since they are less sparse and thus allow a higher recall in sense assignment in context. Or, for another example, imagine that we would like our systems not only to recognise word analogies, but also to explain why the system perceives an analogy. While we can use dense vector spaces to generate/recognise the analogy, we can search for commonalities and differences in the sparse context representation to yield a plausible and readable explanation.

Finally, what will be really needed in the future in order to support adaptive, interactive, iterative and contextualised applications also on the level of language processing are semantic models with a robust representation and are enhanced and improved in the moment new text is processed by the application.

Acknowledgments

Numerous people were involved in the conception of this position paper, which provides the basis for an invited talk at CogALex 2016. Especially, I would like to thank Martin Riedl, Alexander Panchenko, Alfio Gliozzo, Markus J. Hofmann, Michael Zock and many anonymous reviewers from various venues for endless discussions.

References

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Comput. Linguist.*, 36(4):673–721, December.

Luisa Bentivogli, Raffaella Bernardi, Marco Marelli, Stefano Menini, Marco Baroni, and Roberto Zamparelli. 2016. Sick through the semeval glasses: Lessons learned from the evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. *Journal of Language Resources and Evaluation*, 50(1):95–124.

Jon L. Bentley. 1990. K-d trees for semidynamic point sets. In *Proceedings of the Sixth Annual Symposium on Computational Geometry*, SCG '90, pages 187–197, New York, NY, USA. ACM.

Chris Biemann and Martin Riedl. 2013. Text: Now in 2D! A Framework for Lexical Expansion with Contextual Similarity. *Journal of Language Modelling*, 1(1):55–95.

Chris Biemann. 2006. Chinese whispers: An efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*, TextGraphs-1, pages 73–80, New York City, NY, USA.

James R. Curran. 2004. *From distributional to semantic similarity*. Phd thesis, University of Edinburgh. College of Science and Engineering. School of Informatics.

Ferdinand de Saussure. 1966. *Course in General Linguistics*. New York: McGraw-Hill.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407, January.

Ted E. Dunning. 1993. Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics*, 19(1):61–74.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 897–906, Stroudsburg, PA, USA. Association for Computational Linguistics.

Stefan Evert. 2004. *The Statistics of Word Co-occurrences: Word Pairs and Collocations*. Ph.D. thesis, University of Stuttgart.

Stefano Faralli, Alexander Panchenko, Chris Biemann, and Simone Paolo Ponzetto. 2016. Linked disambiguated distributional semantic networks. In *Proceedings of the The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Part II*, pages 56–64, Kobe, Japan.

Tim Feuerbach, Martin Riedl, and Chris Biemann. 2015. Distributional semantics for resolving bridging mentions. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 192–199, Hissar, Bulgaria.

John R. Firth. 1957. *A Synopsis of Linguistic Theory, 1933-1955*. Blackwell, Oxford.

Alfio Gliozzo, Chris Biemann, Martin Riedl, Bonaventura Coppola, Michael R. Glass, and Matthew Hatem. 2013. Jobimtext visualizer: A graph-based approach to contextualizing distributional similarity. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing, TextGraphs-8*, pages 884–890.

Zellig S. Harris. 1951. *Methods in Structural Linguistics*. University of Chicago Press, Chicago.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. COLING*, pages 539–545.

Eduard Hovy. 2010. Distributional semantics and the lexicon (invited talk). In *Proceedings of the 2nd Workshop on Cognitive Aspects of the Lexicon*, page 1, Beijing, China.

Omer Levy and Yoav Goldberg. 2014a. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, MD, USA.

Omer Levy and Yoav Goldberg. 2014b. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems 27*, pages 2177–2185.

Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. 2015. Do supervised distributional methods really learn lexical inference relations? In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 970–976, Denver, CO, USA.

Dekang Lin. 1998. Automatic Retrieval and Clustering of Similar Words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*, pages 768–774, Montreal, QC, Canada.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proc. NIPS*, pages 3111–3119.

George A. Miller and Walter G. Charles. 1991. Contextual Correlates of Semantic Similarity. *Language and Cognitive Processes*, 6(1):1–28.

Tristan Miller, Chris Biemann, Torsten Zesch, and Iryna Gurevych. 2012. Using distributional similarity for lexical expansion in knowledge-based word sense disambiguation. In *Proc. COLING-2012*, pages 1781–1796, Mumbai, India.

Animesh Mukherjee, Monojit Choudhury, Anupam Basu, and Niloy Ganguly. 2008. Modeling the structure and dynamics of the consonant inventories: a complex network approach. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 601–608, Manchester, United Kingdom.

Arvind Neelakantan, 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 (EMNLP)*, pages 1059–1069, Doha, Qatar.

Alexander Panchenko, Dmitry Ustalov, Nikolay Arefyev, Denis Paperno, Natalia Konstantinova, Natalia Loukachevitch, and Chris Biemann. 2016. Human and machine judgements about russian semantic relatedness. In *Proceedings of the 5th Conference on Analysis of Images, Social Networks and Texts (AIST'2016)*, Communications in Computer and Information Science (CCIS), pages 174–183. Springler-Verlag.

Shvaiko Pavel and Jerome Euzenat. 2013. Ontology matching: State of the art and future challenges. *IEEE Transaction on Knowledge and Data Engineering*, 25(1):158–176.

Maria Pelevina, Nikolay Arefiev, Chris Biemann, and Alexander Panchenko. 2016. Making sense of word embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 174–183, Berlin, Germany.

Martin Riedl and Chris Biemann. 2015. A single word is not enough: Ranking multiword expressions using distributional semantics. In *Proc. EMNLP*, pages 2430–2440, Lisbon, Portugal.

Martin Riedl and Chris Biemann. 2016. Unsupervised compound splitting with distributional semantics rivals supervised methods. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 617–622, San Diego, California.

Martin Riedl. 2016. *Unsupervised Methods for Learning and Using Semantics of Natural Language*. Phd thesis, TU Darmstadt, Comp. Sci. Dept.

Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

Gerard Salton, A. Wong, and C. S. Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Mark Steyvers 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.

Kohei Sugawara, Hayato Kobayashi, and Masajiro Iwasaki. 2016. On approximately searching for similar word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2265–2275, Berlin, Germany, August. Association for Computational Linguistics.

György Szarvas, Chris Biemann, and Iryna Gurevych. 2013. Supervised all-words lexical substitution using delexicalized features. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1131–1141, Atlanta, GA, USA.

Ludwig Wittgenstein. 1963. *Tractatus logico-philosophicus. Logisch-philosophische Abhandlung*. Suhrkamp, Frankfurt am Main.

George K. Zipf. 1949. *Human Behavior and the Principle of Least-Effort*. Addison-Wesley, Cambridge, MA.