Embedding a Semantic Network in a Word Space

Richard Johansson and **Luis Nieto Piña**
Språkbanken, Department of Swedish, University of Gothenburg
Box 200, SE-40530 Gothenburg, Sweden
{richard.johansson, luis.nieto.pina}@svenska.gu.se

Abstract

We present a framework for using continuous-space vector representations of word meaning to derive new vectors representing the meaning of *senses* listed in a semantic network. It is a post-processing approach that can be applied to several types of word vector representations. It uses two ideas: first, that vectors for polysemous words can be decomposed into a convex combination of sense vectors; secondly, that the vector for a sense is kept similar to those of its neighbors in the network. This leads to a constrained optimization problem, and we present an approximation for the case when the distance function is the squared Euclidean.

We applied this algorithm on a Swedish semantic network, and we evaluate the quality of the resulting sense representations extrinsically by showing that they give large improvements when used in a classifier that creates lexical units for FrameNet frames.

1 Introduction

Representing word meaning computationally is central in natural language processing. Manual, knowledge-based approaches to meaning representation maps word strings to symbolic concepts, which can be described using any knowledge representation framework; using the relations between concepts defined in the knowledge base, we can infer implicit facts from the information stated in a text: a *mouse* is a *rodent*, so it has prominent *teeth*.

Conversely, data-driven meaning representation approaches rely on cooccurrence patterns to derive a vector representation (Turney and Pantel, 2010). There are two classes of methods that compute word vectors: context-counting and context-predicting; while the latter has seen much interest lately, their respective strengths and weaknesses are still being debated (Baroni et al., 2014; Levy and Goldberg, 2014). The most important relation defined in a vector space between the meaning of two words is *similarity*: a *mouse* is something quite similar to a *rat*. Similarity of meaning is operationalized in terms of geometry, by defining a distance metric.

Symbolic representations seem to have an advantage in describing *word sense ambiguity*: when a surface form corresponds to more than one concept. For instance, the word *mouse* can refer to a *rodent* or an *electronic device*. Vector-space representations typically represent surface forms only, which makes it hard to search e.g. for a group of words similar to the rodent sense of *mouse* or to reliably use the vectors in classifiers that rely on the semantics of the word. There have been several attempts to create vectors representing senses, most of them based on some variant of the idea first proposed by Schütze (1998): that senses can be seen as clusters of similar contexts. Recent examples in this tradition include the work by Huang et al. (2012) and Neelakantan et al. (2014). However, because sense distributions are often highly imbalanced, it is not clear that context clusters can be reliably created for senses that occur rarely. These approaches also lack interpretability: if we are interested in the rodent sense of *mouse*, which of the vectors should we use?

In this work, we instead derive sense vectors by embedding the graph structure of a semantic network in the word space. By combining two complementary sources of information – corpus statistics and network structure – we derive useful vectors also for concepts that occur rarely. The method, which can be applied to context-counting as well as context-predicting spaces, works by decompos-

Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL, pages 1428–1433,
Denver, Colorado, May 31 – June 5, 2015.

ing word vectors as linear combinations of sense vectors, and by pushing the sense vectors towards their neighbors in the semantic network. This intuition leads to a constrained optimization problem, for which we present an approximate algorithm.

We applied the algorithm to derive vectors for the senses in a Swedish semantic network, and we evaluated their quality extrinsically by using them as features in a semantic classification task – mapping senses to their corresponding FrameNet frames. When using the sense vectors in this task, we saw a large improvement over using word vectors.

2 Embedding a Semantic Network

The goal of the algorithm is to *embed* the semantic network in a geometric space: that is, to associate each sense s_{ij} with a *sense embedding*, a vector $E(s_{ij})$ of real numbers, in a way that reflects the topology of the semantic network but also that the vectors representing the lemmas are related to those corresponding to the senses. We now formalize this intuition, and we start by introducing some notation.

For each lemma l_i, there is a set of possible senses $s_{i1}, \ldots, s_{im_i}$ for which l_i is a surface realization. Furthermore, for each sense s_{ij}, there is a *neighborhood* consisting of senses semantically related to s_{ij}. Each neighbor n_{ijk} of s_{ij} is associated with a weight w_{ijk} representing the degree of semantic relatedness between s_{ij} and n_{ijk}. How we define the neighborhood, i.e. our notion of semantical relatedness, will obviously have an impact on the result. In this work, we simply assume that it can be computed from the network, e.g. by picking a number of hypernyms and hyponyms in a lexicon such as WordNet. We then assume that for each lemma l_i, we have a D-dimensional vector $F(l_i)$ of real numbers; this can be computed using any method described in Section 1. Finally, we assume a *distance function* $\Delta(x, y)$ that returns a non-negative real number for each pair of vectors in $\mathbb{R}^D$.

The algorithm maps each sense s_{ij} to a sense embedding, a real-valued vector $E(s_{ij})$ in the same vector space as the lemma embeddings. The lemma and sense embeddings are related through a *mix constraint*: $F(l_i)$ is decomposed as a convex combination $\sum_j p_{ij} E(s_{ij})$, where the $\{p_{ij}\}$ are picked from the probability simplex. Intuitively, the mix variables correspond to the occurrence probabilities of the senses, but strictly speaking this is only the case when the vectors are built using simple context counting. Since the mix gives an estimate of which sense is the most frequent in the corpus, we get a strong baseline for word sense disambiguation (McCarthy et al., 2007) as a bonus; see our followup paper (Johansson and Nieto Piña, 2015) for a discussion of this.

We can now formalize the intuition above: the weighted sum of distances between each sense and its neighbors is minimized, while satisfying the mix constraint for each lemma. We get the following constrained optimization program:

$$\begin{aligned} \underset{E,p}{\text{minimize}} \quad & \sum_{i,j,k} w_{ijk} \Delta(E(s_{ij}), E(n_{ijk})) \\ \text{subject to} \quad & \sum_j p_{ij} E(s_{ij}) = F(l_i) \qquad \forall i \\ & \sum_j p_{ij} = 1 \qquad \forall i \\ & p_{ij} \geq 0 \qquad \forall i, j \end{aligned} \tag{1}$$

The mix constraints make sure that the solution is nontrivial. In particular, a very large number of words are monosemous, and the procedure will leave the embeddings of these words unchanged.

2.1 An Approximate Algorithm

The difficulty of solving the problem stated in Equation (1) obviously depends on the distance function Δ. Henceforth, we focus on the case where Δ is the squared Euclidean distance. This is an important special case that is related to a number of other distances or similarities, e.g. cosine similarity and Hellinger distance. In this case, (1) is a quadratically constrained quadratic problem, which is NP-hard in general and difficult to handle with off-the-shelf optimization tools. We therefore resort to an approximation; we show empirically in Sections 3 and 4 that it works well in practice.

The approximate algorithm works in an online fashion by considering one lemma at a time. It adjusts the embeddings of the senses as well as their mix in order to minimize the loss function

$$L_i = \sum_{jk} w_{ijk} \|E(s_{ij}) - E(n_{ijk})\|^2. \tag{2}$$

The embeddings of the neighbors n_{ijk} of the sense are kept fixed at each such step. We iterate through the whole set of lemmas for a fixed number of epochs or until the objective is unchanged.

Furthermore, instead of directly optimizing with respect to the sense embeddings (which involves $m_i \cdot D$ scalars), the sense embeddings (and therefore also the loss L_i) can be computed analytically if the mix variables $p_{i1}, \ldots, p_{im_i}$ are given, so we have reduced the optimization problem to one involving $m_i - 1$ scalars, i.e. it is univariate in most cases.

Given a sense s_{ij} of a lemma l_i, we define the *weighted centroid* of the set of neighbors of s_{ij} as

$$c_{ij} = \frac{\sum_k w_{ijk} E(n_{ijk})}{\sum_k w_{ijk}}. \quad (3)$$

If the mix constraints were removed, c_{ij} would be the solution to the optimization problem: they minimize the weighted sum of squared Euclidean distances to the neighbors. Then, given the mix, the *residual* is defined

$$r_i = \frac{1}{\sum_j \frac{p_{ij}^2}{\sum_k w_{ijk}}} \left(\sum_j p_{ij} c_{ij} - F(l_i) \right). \quad (4)$$

The vector r_i represents the difference between the linear combination of the weighted centroids and the lemma embedding. Finally, we the sense embeddings for the lemma l_i become

$$E(s_{ij}) = c_{ij} - \frac{p_{ij}}{\sum_k w_{ijk}} r_i. \quad (5)$$

Equations (4) and (5) show the role of the mix variables: if $p_{ij} = 0$, then the sense embedding $E(s_{ij})$ is completely determined by the neighbors of the sense (that is, it is equal to the weighted centroid). On the other hand, if $p_{ij} = 1$, then the sense embedding becomes equal to the embedding of the lemma, $F(l_i)$. To optimize the mix variables $p_{i1}, \ldots, p_{im_i}$ with respect to the loss L_i, basically any search procedure could be used; we found it easiest to use a variant of a simple randomized gradient-free search method (Matyas, 1965).

3 Application to Swedish Data

The algorithm described in Section 2 was applied to Swedish data: we started with lemma embeddings computed from a corpus, and then created sense embeddings by using the SALDO semantic network (Borin et al., 2013). The algorithm was run for a few epochs, which seemed to be enough for reaching a plateau in performance; the total runtime of the algorithm was a few minutes.

3.1 Creating Lemma Embeddings

We created a corpus of 1 billion words downloaded from Språkbanken, the Swedish language bank.[1] The corpora are distributed in a format where the text has been tokenized, part-of-speech-tagged and lemmatized. Compounds have been segmented automatically and when a lemma was not listed in SALDO, we used the parts of the compounds instead. The input to the software computing the lemma embedding consisted of lemma forms with concatenated part-of-speech tags, e.g. *dricka..vb* for the verb 'to drink' and *dricka..nn* for the noun 'drink'. We used the `word2vec` tool[2] to build the lemma embeddings. All the default settings were used, except the vector space dimensionality which was set to 512.

3.2 SALDO, a Swedish Semantic Network

SALDO (Borin et al., 2013) is the largest freely available lexical resource for Swedish. We used a version from May 2014, which contains 125,781 entries organized into a single semantic network. Compared to WordNet (Fellbaum, 1998), there are similarities but also significant differences. Most significantly, SALDO is organized according to the lexical-semantic relation of *association*, not *is-a* as in WordNet. In SALDO, when we go up in the hierarchy we move from specialized vocabulary to the most central vocabulary of the language (e.g. 'move', 'want', 'who'); in WordNet we move from specific to abstract (e.g. 'entity'). Another difference is that sense distinctions in SALDO tend to be more coarse-grained than in WordNet.

Each entry except a special root is connected to other entries, its *semantic descriptors*. One of the

[1] `http://spraakbanken.gu.se`

[2] `https://code.google.com/p/word2vec`

semantic descriptors is called the *primary* descriptor: this is the semantic neighbor that is conceptually most central. Primary descriptors are most often hypernyms or synonyms, but they can also be e.g. antonyms or meronyms, or be in an argument–predicate relationship with the entry. To exemplify, we consider the word *rock*, which has two senses: a long coat and rock music, respectively. Its first sense has the primary descriptor *kappa* 'coat', while for the second sense it is *musik* 'music'.

When embedding the SALDO network using the algorithm in Section 2, the neighbors n_{ijk} of a SALDO sense s_{ij} are its primary descriptor and inverse primaries (the senses for which s_{ij} is the primary descriptor). We did not use any descriptors beside the primary. The neighborhood weights were set so that the primary descriptor and the set of inverse primaries were balanced, and so that all weights sum to 1. If there were N inverse primaries, we set the weight of the primary descriptor to $\frac{1}{2}$ and of each inverse primary to $\frac{1}{2N}$. We did not see any measurable effect of using a larger set of neighbors (e.g. adding connections to second-order neighbors).

3.3 Inspection of Sense Embeddings

Table 1 shows a list of the nearest neighbors of the two senses of *rock*; as we can see, both lists make sense semantically. (A similar query among the lemma embeddings returns a list almost identical to that of the second sense.)

rock-1 'coat'	*rock*-2 'rock music'
syrtut-1 'overcoat'	*punk*-1 'punk music'
kåpa-2 'cloak'	*pop*-1 'pop music'
kappa-1 'coat'	*soul*-1 'soul music'
kavaj-1 'blazer'	*hårdrock*-1 'hard rock'
jacka-1 'jacket'	*hot*-2 'hot jazz'

Table 1: The senses closest to the two senses of *rock*.

A more difficult and interesting use case, where corpus-based methods clearly have an advantage over knowledge-based methods, is to search among the lemmas that have occurred in the corpus but which are not listed in SALDO. Table 2 shows the result of this search, and again the result is very useful. We stress that the list for the first sense would have been hard to derive in a standard word vector space, due to the dominance of the music sense.

rock-1 'coat'	*rock*-2 'rock music'
midjekort 'doublet'	*indie* 'indie'
trekvartsärm '3/4 sleeve'	*indierock* 'indie rock'
spetsbh 'lace bra'	*doo-wop* 'doo-wop'
blåjeans 'blue jeans'	*psykedelia* 'psychedelia'
treggings 'treggings'	*R&B* 'R&B'

Table 2: The unlisted lemmas closest to the two senses of *rock*.

4 Evaluation

Evaluating intrinsically using e.g. a correlation between a graph-based similarity measure and geometric similarity would be problematic, since this is in some sense what our algorithm optimizes. We therefore evaluate extrinsically, by using the sense vectors in a classifier that maps senses to semantic classes defined by FrameNet (Fillmore and Baker, 2009). FrameNet is a semantic database consisting of two parts: first, an ontology of *semantic frames* – the classes – and secondly a lexicon that maps word senses to frames. In standard FrameNet terminology, the senses assigned to a frame are called its *lexical units* (LUs). Coverage is often a problem in frame-semantic lexicons, and this has a negative impact on the quality of NLP systems using FrameNet (Palmer and Sporleder, 2010). The task of finding LUs for frames is thus a useful testbed for evaluating lemma and sense vectors.

To evaluate, we used 567 frames from the Swedish FrameNet (Friberg Heppin and Toporowska Gronostaj, 2012); in total we had 28,842 verb, noun, adjective, and adverb LUs, which we split into training (67%) and test sets (33%). For each frame, we trained a SVM with LIBLINEAR (Fan et al., 2008), using the LUs in that frame as positive training instances, and all other LUs as negative instances. Each LU was represented as a vector: its lemma or sense embedding normalized to unit length.

Table 3 shows the precision, recall, and F-measure for the classifiers for the five frames with most LUs, and finally the micro-average over all frames. In the overall evaluation as well as in four

out of the five largest frames, the classifiers using sense vectors clearly outperform those using lemma vectors. The frame where we do not see any improvement by introducing sense distinctions, PEOPLE_BY_VOCATION, contains terms for professions such as *painter* and *builder*; since SALDO derives such terms from their corresponding verbs rather than from a common hypernym (e.g. *worker*), they do not form a coherent subnetwork in SALDO or subregion in the embedding space.

Frame	P	R	F
ANIMALS	0.741	0.643	0.689
FOOD	0.684	0.679	0.682
PEOPLE_BY_VOCATION	0.595	0.651	0.622
ORIGIN	0.789	0.691	0.737
PEOPLE_BY_ORIGIN	0.693	0.481	0.568
Overall	0.569	0.292	0.386

(a) Using lemma embeddings.

Frame	P	R	F
ANIMALS	0.826	0.663	0.736
FOOD	0.726	0.743	0.735
PEOPLE_BY_VOCATION	0.605	0.637	0.621
ORIGIN	0.813	0.684	0.742
PEOPLE_BY_ORIGIN	0.756	0.508	0.608
Overall	0.667	0.332	0.443

(b) Using sense embeddings.

Table 3: FrameNet lexical unit classification.

5 Conclusion

We have presented a new method to embed a semantic network consisting of linked word senses into a continuous-vector word space; the method is agnostic about whether the original word space was produced using a context-counting or context-predicting method. Unlike previous approaches for creating sense vectors, since we rely on the network structure, we can create representations for senses that occur rarely in corpora. While the experiments described in this paper have been carried out using a Swedish corpus and semantic network, the algorithm we have described is generally applicable and the software[3] can be applied to other languages and semantic networks.

The algorithm takes word vectors and uses them and the network structure to induce the sense vectors. It is based on two separate ideas: first, sense embeddings should preserve the structure of the semantic network as much as possible, so two senses should be close if they are neighbors in the graph; secondly, the word vectors are a probablistic mix of sense vectors. These two ideas are stated as an optimization problem where the first becomes the objective and the second a constraint. While this is hard to solve in the general case, we presented an approximation that can be applied when using the squared Euclidean distance.

We implemented the algorithm and used it to embed the senses of a Swedish semantic network into a word space produced using the skip-gram model. While a qualitative inspection of nearest-neighbor lists of a few senses gives very appealing results, our main evaluation was extrinsic: a FrameNet lexical unit classifier saw a large performance boost when using sense vectors instead of word vectors.

In a followup paper (Johansson and Nieto Piña, 2015), we have shown that sense embeddings can be used to build an efficient word sense disambiguation system that is much faster than graph-based systems with a similar accuracy, and that the mix variables can be used to predict the predominant sense of a word. In future work, we plan to investigate whether the sense vectors are useful for retrieving rarely occurring senses in corpora. Furthermore, since we now evaluated extrinsically, it would be useful to devise intrinsic sense-based evaluation schemes, e.g. a sense analogy task similar to the word analogy task used by Mikolov et al. (2013).

Acknowledgments

This research was funded by the Swedish Research Council under grant 2013–4944, *Distributional methods to represent the meaning of frames and constructions*, and grant 2012–5738, *Towards a knowledge-based culturomics*. The evaluation material used in this work was developed in the project *Swedish FrameNet++*, grant 2010–6013. We also acknowledge the University of Gothenburg for its support of the Centre for Language Technology and Språkbanken.

[3] http://demo.spraakdata.gu.se/richard/scouse

References

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, Baltimore, United States.

Lars Borin, Markus Forsberg, and Lennart Lönngren. 2013. SALDO: a touch of yin to WordNet's yang. *Language Resources and Evaluation*, 47(4):1191–1211.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Christiane Fellbaum, editor. 1998. *WordNet: An electronic lexical database*. MIT Press.

Charles J. Fillmore and Collin Baker. 2009. A frames approach to semantic analysis. In B. Heine and H. Narrog, editors, *The Oxford Handbook of Linguistic Analysis*, pages 313–340. Oxford: OUP.

Karin Friberg Heppin and Maria Toporowska Gronostaj. 2012. The rocky road towards a Swedish FrameNet – creating SweFN. In *Proceedings of the Eighth conference on International Language Resources and Evaluation (LREC-2012)*, pages 256–261, Istanbul, Turkey.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Association for Computational Linguistics 2012 Conference (ACL 2012)*, pages 41–48, Boston, United States.

Richard Johansson and Luis Nieto Piña. 2015. Combining relational and distributional knowledge for word sense disambiguation. In *Proceedings of the 20th Nordic Conference of Computational Linguistics*, Vilnius, Lithuania.

Omer Levy and Yoav Goldberg. 2014. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 171–180, Ann Arbor, United States.

J. Matyas. 1965. Random optimization. *Automation and Remote Control*, 26(2):246–253.

Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2007. Unsupervised acquisition of predominant word senses. *Computational Linguistics*, 33(4):553–590.

Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, USA.

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, October.

Alexis Palmer and Caroline Sporleder. 2010. Evaluating FrameNet-style semantic parsing: the role of coverage gaps in FrameNet. In *Coling 2010: Posters*, pages 928–936, Beijing, China.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.