Towards a Standard Dataset of Swedish Word Vectors

Per Fallgren, Jesper Segeblad, Marco Kuhlmann

Linköping University
581 83 Linköping, Sweden
perfa292@student.liu.se, jesse317@student.liu.se, marco.kuhlmann@liu.se

Abstract

Word vectors, embeddings of words into a low-dimensional space, have been shown to be useful for a large number of natural language processing tasks. Our goal with this paper is to provide a useful dataset of such vectors for Swedish. To this end, we investigate three standard embedding methods: the continuous bag-of-words and the skip-gram model with negative sampling of Mikolov et al. (2013a), and the global vectors of Pennington et al. (2014). We compare these methods using QVEC-CCA (Tsvetkov et al., 2016), an intrinsic evaluation measure that quantifies the correlation of learned word vectors with external linguistic resources. For this propose we use SALDO, the Swedish Association Lexicon (Borin et al., 2013). Our experiments show that the continuous bag-of-words model produces vectors that are most highly correlated to SALDO, with the skip-gram model very close behind. Our learned vectors will be provided for download at the paper's website.

1. Introduction

Word vectors (or word embeddings) 'represent words in a language's vocabulary as points in a d-dimensional space such that nearby words (points) are similar in terms of their distributional properties' (Lin et al., 2015). Recent years have seen a strong demand for such representations. Apart from being interesting in their own right, word vectors can be used as inputs to neural networks, where they have shown to improve accuracy on a large number of natural language processing tasks. Our goal with this work is to provide a useful dataset of word vectors for Swedish.

Many methods for learning word vectors are available today, and the actual learning requires nothing more than suitable training data (usually, tokenized text). The harder task is to choose 'the right' vector set: An informed choice requires a way to evaluate the quality of the learned vectors, but how to do such an evaluation is an open problem and an active area of research. In this paper, we base our choice on QVEC-CCA (Tsvetkov et al., 2016), an intrinsic measure that quantifies the correlation of the learned vectors with properties obtained from existing linguistic resources. For this we propose to use the 'supersenses' of SALDO, the Swedish Association Lexicon (Borin et al., 2013). For training the vectors, we use a corpus consisting of approximately 220M tokens, exclusively obtained from Språkbanken's *Göteborgsposten* data set. We produce vectors for a vocabulary of 192k words.

The next section presents different word embedding methods, with a focus on the methods that we compare in this paper. We then present our adapted version of QVEC-CCA in Section 3. Sections 4 and 5 describe the setup and the results of our experiments, and Section 6 discusses the implications of these results for our recommendation of a standard dataset of Swedish word vectors. Section 7 concludes the paper with some ideas for future work.

2. Word Embedding Methods

Standard approaches for generating word embeddings are based on the idea that words that occur in similar contexts should be close to each other as vectors (Harris, 1954).

Well-known methods for word embeddings include Latent Semantic Analysis (LSA) (Dumais et al., 1988), in which the context of a target word is defined as the document that it occurs in, and the Hyperspace Analogue to Language (HAL) (Lund and Burgess, 1996), which is based on co-occurrence statistics about the words immediately before and after the target word. In both of these methods, the dimensionality of the resulting vectors grows with the number of distinct contexts (documents or context words) that the target words appear in, which can become impractical when the methods are applied to large amounts of text. In more recent word embedding methods, the number of dimensions can be fixed prior to training.

There are several approaches to generating high-quality word vectors with fixed size. For example, Singular Value Decomposition (SVD) can be used to reduce the dimensionality of a matrix generated with LSA or HAL. Random Indexing (RI) (Sahlgren, 2005) is another technique where the vectors are of fixed size. In this study, we use three different methods for creating word embeddings: the continuous bag-of-words (CBOW) and the skip-gram model (SGNS) implemented in word2vec (Mikolov et al., 2013a),[1], and GloVe (Pennington et al., 2014). The following paragraphs briefly present these methods.

2.1 Word2vec

The basic idea behind the models in word2vec (Mikolov et al., 2013a; Mikolov et al., 2013b) is to learn word vectors in the context of a prediction task: in the CBOW model, the task is to predict the target word from the words in the surrounding linear window; in the SGNS model, it is the other way around. The structure of the two models is a shallow neural network; the learned word vectors are the outputs of the hidden layer of that network. A more thorough explanation of the word2vec models can be found in the original papers by Mikolov et al. (2013a) and Mikolov et al. (2013b), as well as in the subsequent analysis of Goldberg and Levy (2014).

[1] Both models were used with negative sampling.

2.2 GloVe

The GloVe algorithm is introduced by Pennington et al. (2014). In their paper they split the field of word vector learning into two families. The first one is *global matrix factorization methods*; LSA is an example of this. These perform well in capturing global aspects of a corpus, but not so well in analogy tasks. The other family is called *local context window methods*; this family includes the two word2vec models. These methods perform well in analogy tasks, but do not capture the global statistics equally well. GloVe is designed to combine the advantages of the two approaches. It is a log-bilinear model with a weighted least-squares objective that tries to adjust the entries of the vectors so that their dot product approaches the probability of the words co-occurring with each other.

3. Evaluation of Word Embeddings

How do we know which of the available methods for learning word vectors performs best for a given dataset? The most common way to evaluate the quality of word vectors is to test them on word similarity tasks (Finkelstein et al., 2002; Bruni et al., 2012; Hill et al., 2015) or analogy tasks (Mikolov et al., 2013a). Unfortunately, such datasets are not available for Swedish, and while one could build a new such dataset, it is a hard task. Apart from annotation being expensive, the design of such a dataset is difficult. This is partly due to the problems associated with the datasets currently used for word similarity tasks, such as low correlations with downstream tasks, the problem of accounting for polysemous words, and defining the types of similarities to include (Faruqui et al., 2016).

In this paper we use QVEC (Tsvetkov et al., 2015; Tsvetkov et al., 2016), an intrinsic evaluation method that can make use of already existing resources, and has been shown to correlate well with performance in downstream tasks. The method essentially measures the correlation between the vector set and a manually crafted set of 'linguistic' vectors. The linguistic vectors can be understood as term frequency vectors normalised to probabilities, where the terms express some linguistic property.

Tsvetkov et al. (2015) used WordNet supersenses as dimensions in the linguistic vectors, and SemCor (A. Miller et al., 1993) to extract sense frequencies. As WordNet is not available for Swedish, we choose to replace this resource by SALDO, the Swedish Association Lexicon (Borin et al., 2013). In SALDO, each word has a so-called primary descriptor, which is a more abstract or general description of the word. As such, the structure of SALDO can be viewed as a tree, with the primary descriptor being the parent of the words it describes. At the top of the tree, there are 41 semantic primitives[2] which are words that cannot be described by a more general one. These primitives act as 'supersenses', where all words that are subsumed by a specific primitive, directly or indirectly, have that sense as its supersense. Instead of SemCor, we use version 3.0 of the Stockholm–Umeå corpus (SUC)[3], which has been automatically annotated with word senses from SALDO, to extract sense frequencies for all words occuring more than 5 times in both corpora.

In its orginal form, QVEC is limited to the comparison of vectors of the same dimensionality, and is not invariant to linear transformations of the embedding's basis. In follow-up work, Tsvetkov et al. (2016) presented QVEC-CCA which solves both of these problems, while still showing medium to high correlations with performance in downstream tasks. QVEC-CCA uses canonical correlation analysis to measure the correlation between the word embedding matrix and the linguistic properties matrix, and produces a single score in the interval $[-1, +1]$, where -1 means 'perfect negative correlation' and $+1$ means 'perfect positive correlation'.

4. Data and Experiments

With a measure for comparing word embedding methods at hand, we trained different models in order to identify the model best aligned to the linguistic dimensions of SALDO.

4.1 Data

Our vector sets were trained on a corpus with texts from the Swedish newspaper *Göteborgsposten* (GP) from the years 2001 to 2013. This material, which is freely available from Språkbanken (the Swedish Language Bank)[4], was chosen primarily because of its relatively large size (approximately 1.5 times the number of tokens in Swedish Wikipedia) and the fact that it consists of coherent and curated text. While Swedish corpora of larger size are available, these are crawled from uncurated sources such as web forums, which we anticipated would impede the quality of the word vectors. The training data consisted of 17,397,223 sentences with 220,290,482 tokens; the total size of the data was 1.4 GB. With a frequency threshold of 25 the produced vector sets consisted of 192,250 word vectors. The corpus, which comes in an XML format with automatically assigned parts-of-speech, dependency relations and other linguistic annotations, was reduced to the bare tokenised text. Each sentence was lowercased.

4.2 Models

We produced a total of 45 vector sets, 15 for each of the three word embedding methods (CBOW, SGNS, GloVe). The study consisted of two phases: In the first phase, for each method we created a total of 10 vector sets with context window sizes from 2 to 10, with a step size of 2 and the default 5 iterations. Half of the models were created with a dimensionality of $d = 50$, half of the models with a dimensionality of $d = 300$. In the second phase of the study, for each method we created 5 additional vector sets with a dimensionality of $d = 300$, with the window size fixed to 10. This was done by increasing the amount of iterations from 10 to 50 with a step size of 10. Apart from window size and number of iterations, we used default settings for the respective methods.

[2] Borin et al. (2013) mentions 43 semantic primitives, but we were only able to identify 41 semantic primitives in the release that we used for our experiments.

[3] https://spraakbanken.gu.se/

[4] https://spraakbanken.gu.se/

Model	2	4	6	8	10
CBOW	0.1746	0.1766	0.1785	**0.1798**	0.179
SGNS	0.1856	0.1898	0.1909	0.1908	**0.1916**
GloVe	0.1377	0.1495	0.1515	0.1555	**0.1566**

(a) $d = 50$

Model	2	4	6	8	10
CBOW	0.3428	0.3476	0.3496	0.3505	**0.3516**
SGNS	0.3479	0.3540	**0.3555**	0.3542	0.3536
GloVe	0.2637	0.2783	0.2849	0.2879	**0.2908**

(b) $d = 300$

Table 1: Impact of window size on QVEC-CCA scores for models with $d = 50$ and $d = 300$ and 5 iterations. Top score for each model in bold.

5. Results

In this section we present the results of our experiments. All scores are the QVEC-CCA scores of the learned vector set, computed as described in Section 3.

Effect of Larger Window Sizes Table 1 shows the effect of an increased window size on the QVEC score. We first discuss the results for $d = 300$, which are visualised in Figure 1. As we see, there is a steady increase in performance for each step size, with a top result of 0.3516 for a window size of 10. In the case of SGNS, we see that the QVEC score initially increases (until window size 6) but then decreases. Finally, for GloVe, there is also a steady increase in performance for each step size, with a top result of 0.2908 at a window size of 10. For $d = 50$, we see a similar trend: Generally, there is an overall increase in performance corresponding with the window size, with the highest score for window size 10. For CBOW the top performance of 0.1798 is observed at a window size of 8. The top performance of SGNS was 0.1916, observed at a window size of 10. Similarly, the top result for GloVe was 0.1566, observed at a window size of 10.

Effect of More Iterations Table 2 shows the result for the second round of experiments, in which we fixed the dimensionality at $d = 300$ and the window size at 10 but increased the number of iterations. Neither of the models generated a steady increase or decrease in performance when adjusting the amount of iterations. The top score was observed for CBOW (0.3570, 40 iterations)[5], outperforming the result of SGNS in the first phase. The top score for SGNS was 0.3537 (10 iterations). Finally, the top score for GloVe was 0.2911 (20 iterations).

6. Discussion

Effect of Larger Window Sizes Looking at Table 1 and Figure 1, it is clear that that QVEC-CCA prefers larger window sizes to smaller ones. With the exception of SGNS at $d = 300$, which had a top result at a window size of 6, and CBOW at $d = 50$, which had a top result at a window size of 8, the models all performed best at a window size of 10.

[5]Using a 2,4 GHz Intel Core i5 MacBook Pro the training time for the top performing dataset was 7 hours and 11 minutes.

Model	10	20	30	40	50
CBOW	0.355	0.3557	0.3566	**0.3570**	0.3565
SGNS	**0.3537**	0.3534	0.3517	0.3524	0.3525
GloVe	0.2901	**0.2911**	0.2898	0.2893	0.2885

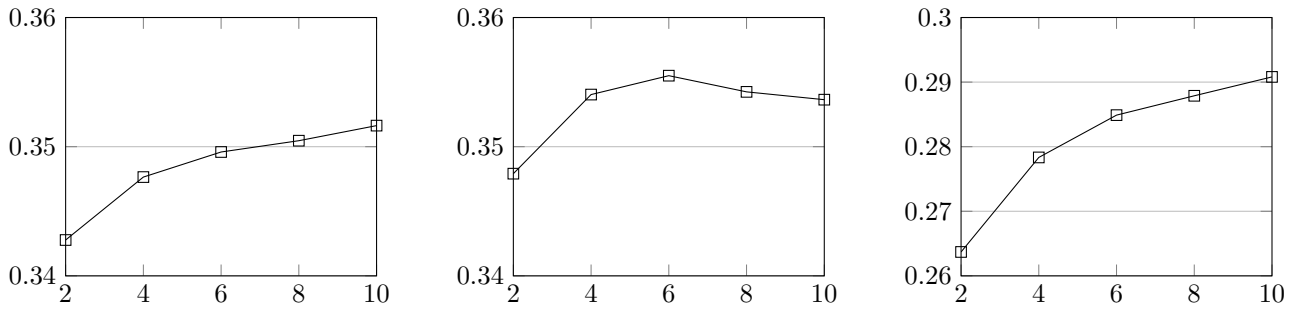
Table 2: Impact of number of iterations on QVEC-CCA score for models with $d = 300$ and window size 10. Top score for each model in bold.

Although it would have been even more conclusive if the two exceptions had been generated by the same model, it seems clear that higher window sizes lead to higher scores. Similar observations have been made in previous studies; in particular, the positive correlation between window size and quality relates to the result of Pennington et al. (2014): In the second diagram of their Figure 2 they show an increase in score that slowly stagnates when reaching larger window sizes. The prominent observation in both studies is the high increase in score when choosing a window size of 4 instead of 2, and that the graphs flatten out when choosing window sizes of higher values. The consistency observed in these results corroborates the validity of our evaluation method. QVEC-CCA is a fairly new evaluation method, and while it has been shown to correlate well with performance in downstream tasks in English when used together with WordNet, one cannot be sure how well it translates to Swedish and SALDO. Our interpretation of our results is that our evaluation measure is valid. Apart from the previously mentioned aspects of consistency, it is quite intuitive that a higher number of dimensions should lead to better correlation with linguistic information, and hence to higher scores, assuming that the evaluation method is valid. This is the case for all the models explored, where the 300-dimensional vectors outperform the 50-dimensional ones.

Effect of more iterations In Table 2, we see that the top result is achieved by the CBOW model trained with 40 iterations. While SGNS yielded the top-performing vector set of the first phase, it does not produce as high results in the second phase. The top score of SGNS is however still close behind; GloVe on the other hand does not seem to perform as well. Interestingly, with the exception of CBOW, an increase of iterations does not lead to an increase in score, which flattens out rather early, and the small fluctuations that are present are most likely from coincidence rather than a difference in quality. Nevertheless, there is still sufficient evidence that the evaluation method is valid, as mentioned in the previous section, and that the vectors generated by CBOW make a useful set of word embeddings that we can recommend for further studies.

7. Conclusion

This study compared three methods for creating word embeddings, with the main purpose of producing a useful set of Swedish word vectors. When using our evaluation method, which combines QVEC-CCA and SALDO, the highest-scoring vector set is created with the CBOW algorithm at $d = 300$ and 40 iterations. This vector set should provide a useful off-the-shelf set of word vectors for future work in Swedish natural language processing.



(a) CBOW (b) SGNS (c) GloVe

Figure 1: Impact of increasing window size (x-axes) on model performance as measured in terms of QVEC-CCA score (y-axes). All models were trained with $d = 300$ and the default number of iterations.

The Göteborgsposten corpus was used in this study because of its substantial size and its coherent structure with respect to formality, language, terminology, and grammar. One could certainly concatenate different corpora for even greater size. However, if one is interested in other aspects than the size or domain of the corpus, then there are alternatives. There is a long list of Swedish corpora to choose from from the Swedish Language Bank. For a specific domain, one will probably create custom corpora, and the three methods that we have investigated in this study may not perform equally well on these. There are also alternatives with respect to how the data can be preprocessed, normalised, and augmented. For example, Trask et al. (2015) show how incorporating part-of-speech tags can greatly reduce issues regarding ambiguous words.

Our vector set is available for download at http://www.ida.liu.se/divisions/hcs/nlplab/swectors/.

References

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. In *Human Language Technology: Proc. of a Workshop Held at Plainsboro, New Jersey, March 21–24*, pages 303–308.

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

Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012. Distributional semantics in technicolor. In *Proc. of ACL*, pages 136–145.

Susan T. Dumais, George W. Furnas, Thomas K. Landauer, Scott Deerwester, and Richard Harshman. 1988. Using latent semantic analysis to improve access to textual information. In *Proc. of SIGCHI Conference on Human Factors in Computing Systems*, pages 281–285.

Manaal Faruqui, Yulia Tsvetkov, Pushpendre Rastogi, and Chris Dyer. 2016. Problems with evaluation of word embeddings using word similarity tasks. In *Proc. of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 30–35.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20:116–131.

Yoav Goldberg and Omer Levy. 2014. Word2vec explained: Deriving Mikolov et al.'s negative-sampling word-embedding method. *CoRR*, abs/1402.3722.

Zellig S. Harris. 1954. Distributional structure. *Word*, 10:146–162.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41:665–695.

Chu-Cheng Lin, Waleed Ammar, Chris Dyer, and Lori Levin. 2015. Unsupervised POS induction with word embeddings. In *Proc. of NAACL-HLT*, pages 1311–1316.

Kevin Lund and Curt Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*, 28:203–208.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proc. of Conference on Advances in Neural Information Processing Systems*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proc. of EMNLP*, pages 1532–1543.

Magnus Sahlgren. 2005. An introduction to random indexing. In *Proc. of Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering*.

Andrew Trask, Phil Michalak, and John Liu. 2015. sense2vec – A fast and accurate method for word sense disambiguation in neural word embeddings. *CoRR*, abs/1511.06388.

Yulia Tsvetkov, Manaal Faruqui, Wang Ling, Guillaume Lample, and Chris Dyer. 2015. Evaluation of word vector representations by subspace alignment. In *Proc. of EMNLP*, pages 2049–2054.

Yulia Tsvetkov, Manaal Faruqui, and Chris Dyer. 2016. Correlation-based intrinsic evaluation of word vector representations. In *Proc. of 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 111–115.