Evaluating Embeddings using Syntax-based Classification Tasks as a Proxy for Parser Performance

Arne Köhn
Deparment of Informatics
Universität Hamburg
koehn@informatik.uni-hamburg.de

Abstract

Most evaluations for vector space models are semantically motivated, e.g. by measuring how well they capture word similarity. If one is interested in syntax-related downstream applications such as dependency parsing, a syntactically motivated evaluation seems preferable. As we show, the choice of embeddings has a noticeable impact on parser performance. Since evaluating embeddings directly in a parser is costly, we analyze the correlation between the full parsing task and a simple linear classification task as a potential proxy.

1 Introduction

Many complex tasks in NLP are solved using embeddings as additional features. In some pipelines, pre-trained embeddings are used as-is, in others they are learned as an integral part of training the pipeline (examples for this would be e. g. RNNLMs(Mikolov et al., 2010) or parsers that learn their own embeddings (e. g. Chen and Manning (2014))). We focus on the first type. If we want to run a system that can be enhanced using pre-trained embeddings, the question arises which embedding actually works best.

Since it is computationally infeasible to evaluate all embeddings on all pipelines, usually simple tasks are used to demonstrate the strengths of embeddings and an embedding for use in a pipeline is picked based on these proxy tasks. Most of these tasks are semantically motivated and English dominates as language of choice for the tasks. Last year, we proposed a more syntactically motivated evaluation task *syneval*, which uses morpho-syntactic information across a variety of languages (see Section 2).

Morphological information helps syntactic parsers, but usually there is no gold-standard information available during parsing (with the exception of parser evaluation). Using embeddings that are good predictors of the missing morphological information should alleviate the problem of missing morphology.

It is reasonable to assume that the classification problems in syneval are a good proxy for syntax-related tasks because they describe how well an embedding is able to capture morphological information which is helpful to the parser. To test this assumption, we evaluate the performance of RBGParser (Lei et al., 2014; Zhang et al., 2014) using different embeddings as additional features. The parser performance using a specific embedding should then reflect the embedding's performance on the classification tasks. We know that embeddings yield only marginal improvements if the parser also has access to gold standard morphological information but benefits significantly if no morphological information is present (Lei et al., 2014; Köhn et al., 2014). Therefore, we experiment with stripping the information that is used as classification target in syneval.

2 Syneval

In previous work, we proposed to make use of treebanks to extract simple syntactic evaluation tasks (Köhn, 2015) but somehow didn't assign a catchy name. We now make up for this and call this approach *syneval* throughout this paper. For a given syntactic feature type F (e.g. tense or case), a classification task is created as follows: Let W be the set of words forms and $V \subseteq \mathbb{R}^n$ the vector space of a given embedding $W \rightarrow V$. Using a treebank where some words[1] are annotated with

[1] Some words are not annotated with features of certain types, e. g. nouns are usually annotated without tense markers.

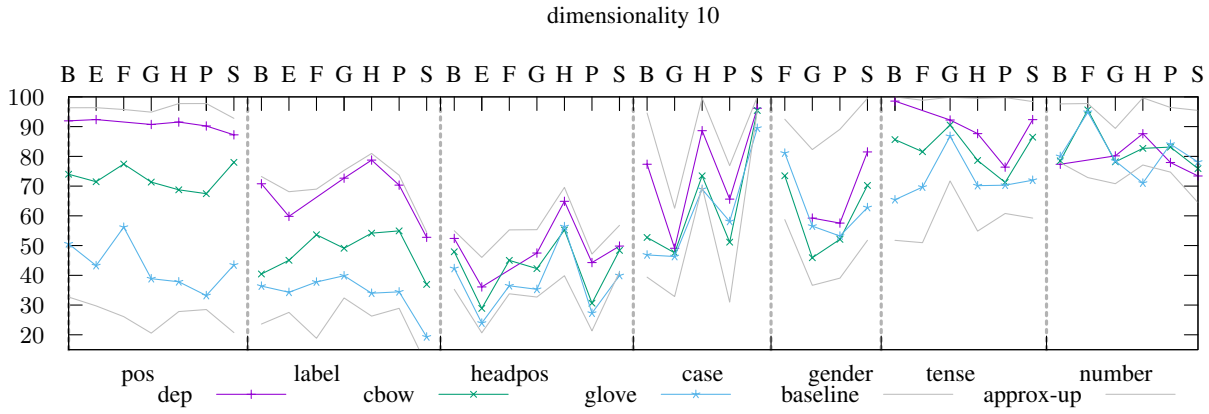

Figure 1: Accuracies for the proxy task *syneval* evaluated on **B**asque, **E**nglish, **F**rench, **G**erman, **H**ungarian, **P**olish, and **S**wedish. Note that dep was not evaluated on French. All results are from Köhn (2015). Results for embeddings not discussed in this paper have been omitted.

a syntactic feature of the given type $f \in F$, we combine the syntactic information with the word vector $v \in V$ of the word, obtaining a pair (v, f) for each word. In other words, we perform an inner join of the embedding and the syntactic annotation on the word and project on V and F.

Note that there is no functional dependence $W \rightarrow F$ because the same word form can have different syntactic features depending on the context. Therefore, there is also no functional dependence between the word vectors and the syntactic features.

A linear classifier is trained on (v, f) pairs to predict the syntactic feature given the word embedding. If the classifier yields a high accuracy, the embeddings encode structure with respect to the syntactic feature type. The vector space is partitioned by the classifier into convex polytopes which each represent one syntactic feature (such as NN for the PoS classification task) and if the classifier has a high accuracy, these polytopes accurately describe the features. Since the classifier does not use any context features and a word vector can be paired with different syntactic features, the upper bound for classification accuracy can be approximated by classification based on the word form. The lower bound is the majority baseline, i.e. using no features at all.

The syntactic features used in syneval are: *pos* (the PoS of the word), *label* (the dependency label), *headpos* (the PoS of the word's head) as well as the morphological features *case*, *gender*, *tense*, and *number*. Syneval results for a selected set of embeddings are depicted in Figure 1.

Syneval has several advantages over other embedding evaluation methods: First of all, it uses several languages instead of being centered on English. It does not need manually generated data such as similarity judgments as the treebanks used for evaluation have already been built for other purposes. Syneval covers much more lexical items than other evaluations: SimLex-999 (Hill et al., 2015, one of the larger word similarity corpora) contains 1030 word forms whereas syneval performs an evaluation on nearly 30.000 word forms for English.

3 Data and Embeddings

To limit the amount of computation, we select four languages out of the seven evaluated by Köhn (2015), namely Basque, German, Hungarian, and Swedish, and three embeddings out of six. Even with these reductions, the experiments for this paper needed about 500 CPU-days. All experiments are performed on data from the SPMRL 2014 (Seddah et al., 2014), using the full training set for each language and the dev set for evaluation.

The embeddings are taken from Köhn (2015). We use the ten-dimensional embeddings, as the differences between the approaches are more pronounced there, and we can be sure that the parser does not drain in high dimensional features (Lei et al. (2014) used 25 and 50 dimensional vectors). Again, we limit the number of embeddings to three: skip-gram using dependency contexts (*dep*, (Levy and Goldberg, 2014)), *GloVe* (Pennington et al., 2014), and word2vec using *cbow* (Mikolov et al., 2013). In the syneval evaluation, dep performed best, GloVe worst, and cbow in between (see Fig-

		dep	cbow	GloVe	none			dep	cbow	GloVe	none
Basque	+all	89.77	89.13	89.79	**90.07**	Hungarian	+all	**88.66**	88.54	88.24	88.39
	-case	89.43	88.12	88.30	88.41		-case	87.52	87.37	87.46	87.10
	-tense	89.81	88.80	89.71	**89.97**		-tense	**87.50**	87.41	87.59	87.26
	-number	89.88	88.86	89.26	89.86		-number	87.59	87.34	87.60	87.07
	-PoS	**88.22**	86.39	87.94	87.99		-PoS	**85.97**	85.72	85.80	85.42
	-all	**85.51**	80.99	81.68	79.24		-all	**81.18**	78.69	78.24	76.08
German	+all	94.87	94.67	94.89	94.82	Swedish	+all	85.17	85.06	84.83	85.17
	-case	94.38	94.20	94.40	94.42		-case	85.20	84.94	84.97	85.15
	-tense	94.87	94.66	94.81	94.76		-tense	84.94	84.94	**85.27**	85.15
	-number	94.84	94.60	94.77	94.83		-number	85.07	84.81	85.06	**85.19**
	-PoS	91.24	90.15	91.15	91.22		-PoS	**79.53**	78.21	78.65	78.68
	-all	**88.26**	86.68	87.72	87.35		-all	**76.55**	73.79	73.41	71.11

Table 1: Unlabeled parsing accuracies using different embeddings as well as no embeddings with varying amounts of gold-standard morphological information available. Results better than the second best by a margin of at least .1 are highlighted.

ure 1). For some tasks, GloVe barely outperforms the majority baseline, i.e. it does not contain much information that can be extracted with a linear classifier.

4 Using Embeddings in a Parser

To evaluate the benefit of using the embeddings mentioned in the previous section in a parser, the parser needs to fulfill several requirements: The parser needs to work both with and without embeddings, it needs to use pre-trained embeddings, and it should make use of morphological features. If all these requirements are fulfilled, it is possible to measure the benefit of different embeddings as well as using embeddings at all, and whether morphological information supersedes such benefits.

Based on the requirements, we chose RBGParser for the evaluation. RBGParser uses embeddings for scoring edges using low-rank tensors. To score edges, the function uses the embedding, form, lemma, pos, and morphological information of the words as well as arc length and direction. In addition to the low-rank tensor, it uses the same features as TurboParser (Martins et al., 2013) as well as some features encoding global properties. Both components are weighted using a hyperparameter which we keep fixed at the default value. Since the embeddings are only used in the tensor component, the quality of the embeddings only affect this component.

Nevertheless, we chose to use the whole parser including all features instead of just measuring the impact on the low-rank tensor component because it is possible that improvements in this component don't translate to an improvement of the whole parser.

5 Experiments

The basic idea is as follows: If an embedding encodes a morphological feature well, it should be a good drop-in replacement of that feature. Therefore, if we strip a morphological feature from the data, using a well-performing embedding should yield higher parsing accuracies than using a worse performing one.

We use the following setups with each embedding (as well as without embeddings):

- no case information (-case)
- no tense information (-tense)
- no number information (-number)
- no PoS information (-PoS)
- morphology (including PoS) completely stripped (-all)
- all gold standard information as-is (+all)

We train RBGParser on the gold standard for each language using the settings mentioned above, i. e. stripping the morphological information corresponding to the setting from both training and test data. For each setting and language, we trained the parser in for modes: Without Embeddings, with dep, with GloVe, and with cbow. The resulting accuracies are listed in Table 1. No embedding is able to provide a benefit to the parser with complete

gold-standard annotations (the +all rows), which is consistent with previous findings.

Even when stripping morphological information, the embeddings only yield relevant improvements with respect to not using embeddings in -PoS and -all settings. In both these cases, dep clearly outperforms the other embeddings, which is consistent with the syneval results (Figure 1). In contrast, cbow, which performs better than GloVe in syneval, yields worse results than GloVe on average. Both differences are significant (two-sided sign test; dep $\neq$ glove: $p < 0.05$; GloVe $\neq$ cbow: $p < 0.01$). The absolute difference between dep and GloVe is much larger than the absolute difference between GloVe and cbow. The difference between GloVe and cbow is especially striking in the -PoS case where cbow outperforms GloVe by a large margin in syneval but is consistently beaten in the parsing task for every language, even those where cbow outperforms GloVe in the +all case.

Stripping a single morphological feature (other than PoS) has little impact on parsing accuracy. On the other hand, stripping all morphological information leads to much worse accuracies than just parsing without PoS. This hints at some redundancy provided by the morphological annotations.

6 Conclusions

Syneval and the downstream parser evaluation both reveal large differences between the different embeddings. dep outperforms all other embeddings in syneval for all tasks except number-Polish and number-Spain and also is most helpful to the parser with considerable margin. The embeddings are only consistently helpful to the parser if no PoS-tags are provided.

Despite the consistency of the dep result between syneval and the parsing task, our findings are inconclusive overall. On the one hand, the by far best performing approach on the proxy task also performed best for parsing, on the other hand cbow performed worse than GloVe in the parsing task despite performing better in the proxy task. This indicates that there is helpful information encoded that is not captured by the proxy task, but which interestingly can not be realized when parsing with full gold-standard morphological annotation.

Code and data for this work is available under `http://arne.chark.eu/repeval2016`.

References

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar, October. Association for Computational Linguistics.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695, Sep.

Arne Köhn, U Chun Lao, AmirAli B Zadeh, and Kenji Sagae. 2014. Parsing morphologically rich languages with (mostly) off-the-shelf software and word vectors. In *Proceedings of the 2014 Shared Task of the COLING Workshop on Statistical Parsing of Morphologically Rich Languages*.

Arne Köhn. 2015. What's in an embedding? analyzing word embeddings through multilingual evaluation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2067–2073, Lisbon, Portugal, September. Association for Computational Linguistics.

Tao Lei, Yu Xin, Yuan Zhang, Regina Barzilay, and Tommi Jaakkola. 2014. Low-rank tensors for scoring dependency structures. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1381–1391, Baltimore, Maryland, June. Association for Computational Linguistics.

Omer Levy and Yoav Goldberg. 2014. 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, Maryland, June. Association for Computational Linguistics.

André Martins, Miguel Almeida, and Noah A. Smith. 2013. Turning on the turbo: Fast third-order non-projective turbo parsers. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 617–622, Sofia, Bulgaria, August.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of INTERSPEECH-2010*, pages 1045–1048.

Tomáš Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Jeffrey Pennington, 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 (EMNLP), pages 1532–1543, Doha, Qatar, October. Association for Computational Linguistics.

Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho Choi, Matthieu Constant, Richárd Farkas, Iakes Goenaga, Koldo Gojenola, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiorkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, Alina Wróblewska, and Eric Villemonte de la Clérgerie. 2014. Overview of the SPMRL 2014 shared task on parsing morphologically rich languages. In *Notes of the SPMRL 2014 Shared Task on Parsing Morphologically-Rich Languages*, Dublin, Ireland.

Yuan Zhang, Tao Lei, Regina Barzilay, Tommi Jaakkola, and Amir Globerson. 2014. Steps to excellence: Simple inference with refined scoring of dependency trees. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 197–207, Baltimore, Maryland, June. Association for Computational Linguistics.