Evaluation of Word Embeddings for Sequence Tagging

A Anonymous **B Anonymous**

Abstract

XXX

1 Introduction

Word embedding learning methods are the new generation of distributional semantics models. As with other learning methods, it is well known that the performance of machine learning algorithms heavily depends on their parameters optimization, the training data used and the applications they target. In this paper, we perform an extensive evaluation of three word embedding approaches under the same experiment conditions, and evaluate them in different sequence tagging tasks.

Expected contributions:

- Fair comparison of different word embedding algorithms
- Influence of word embeddings in the sequence tagging tasks with semi-supervised settings.
- Complete evaluation of word embeddings in sequence tagging tasks.
- Multi-word expressions identification using word embeddings.

2 Related Work

In this section, we describe the three word embedding approaches we compared in the paper.

Explain what are word embeddings Word embeddings models are the new generation of distributional semantics models (DSMs), where the vectors weights are set to optimally predict the contexts in which the corresponding word tend to appear.

Describe each of the following word embeddings approaches

- Glove (Pennington et al., 2014)
- word2vec (Mikolov et al., 2013)
- Brown cluster (Brown et al., 1992)
- Pre-trained word embeddings (Turian et al., 2010a)

The mentioned methods were chosen because they are recent state-of-the-art word embedding learning and because their their software is available.

3 Word Representations

We test the effectiveness of different word embeddings approaches on four different sequence labeling task, which are the following,

It is well known that the choice of a corpora have an important effect in the final accuracy of machine learning algorithms. Thus, we select different corpora to learn the word embedding vectors (Table 3). The main reason of choosing these data set is that they are publicly available.

Data set	Size	Words
One Billion	4.1GB	0.8 billions
UMBC	48.1GB	3 billions
Latest wikipedia dump	49.6GB	3 billions
Twiter		

Table 1: Corpus used to learn word embeddings

3.1 Word Embedding Learning Algorithms

- Glove (Pennington et al., 2014)
- Skig-gram (Mikolov et al., 2013)
- CBOW (Mikolov et al., 2013)
- Brown cluster (Brown et al., 1992)
- Neural language model (Turian et al., 2010a)

3.2 Experimental Setup

3.3 Preprocessing

In order to make the comparison of different word embedding approaches across different applications, we applied the same preprocessing to the data sets used. The preprocessing pipeline consist of a sentence splitter, a tokenizer, a POS-tagger and a lemmatizer. The pipeline is built with the UIMA architecture and the DKPro NLP tools.

3.4 Parameters

The performance of each approach heavily depends on their parameters optimization. In an ideal machine learning setup, grid search or random search would be applied in order to search for the best hyperparameters, for each approach. But, that is too time taken. Instead, we look for the shared parameters along the three approaches and vary these parameters deterministically? The rest of the parameters (the ones that are unique for each approach) are set to their optimal reported values. The parameters that we vary are:

- **Word vector size**: 25, 50, 100, 200, 300, 600.

- **Context window size**: 5, 10, 15.

Brown clustering: Number of clusters

4 Sequence Tagging Tasks

We evaluate different learning approaches of word embeddings in four different sequence tagging tasks: POS tagging, chunking, MWE identification, and name entity recognition. Because we can either update pre-trained word embeddings during training or not, through the evaluation, we want to answer the following questions:

- How well do different word embeddings perform in all tasks when supervised fine-tuning is *not* performed?

- How well do different word embeddings perform in all tasks when supervised fine-tuning is performed?

- How does the size of labeled training data affect the experimental results?

- How well do the word embeddings perform for unknown words?

- How do the key parameters of each word learning algorithms affect the experimental results?

For each task, we feed learned embeddings into the graph transformer trained with sentence tag criterion (Turian et al., 2010b). The graph transformer is equivalent to CRF, if we do not update word embeddings. For all tasks, we train Graph transformer with pre-trained word embeddings in the following two settings:

- CRF with conventional features.

- Graph transformer *does not* fine tune embeddings during training.

- Graph transformer fine tunes the embeddings during training.

For each task, we split the data into a training set, validation set, and a test set. The hyper parameters are tuned on the validation set with random search (Bergstra and Bengio, 2012). To be fair, for each model, we randomly choose 100 hyper parameter combinations and pick up the best one based on its performance on the validation set. Then each model is evaluated in a semi-supervised setting. We start with training models on 10% of the training data, and evaluate them on the test dataset. Then we incrementally add another 10% of the training data and evaluate them until all training data is used. We adopt per-word F1 scores as the evaluation metric for all tasks except POS tagging. We keep using per-word accuracy for POS tagging. In order to evaluate model performance on unknown words, we report also the average F1 scores for the words that do not occur in the training set.

In order to assess the influence of the key parameters of each word learning algorithms, we evaluate all embedding based models with varying key parameters on the full training set.Do we evaluate also combinations of the key parameters ,e.g. we measure model performance by incrementally augmenting the size of the training set at the same time as the word vector size.?

4.1 POS tagging

Almost the setting as (Collobert et al., 2011), except adding one more test set.

4.1.1 Option 1

Training set: 0-18 of WSJ.
Validation set: 19-21 of WSJ.
Test set: 22-24 of WSJ, and English Web Treebank. We report model performances on these two test sets respectively.
Feature space: the same set as in (Collobert et al., 2011)

4.1.2 Option 2

Use the experimental setting in (Owoputi et al., 2013) for twitter POS tagging. All word embeddings will be learned from the twitter corpus provided by Scott.

4.2 Chunking

The same setting as (Turian et al., 2010b)

Training set: WSJ train set.
Validation set: Randomly sampled 1000 sentences from the train set for development.
Test set: CoNLL2000 test set.
Feature space: the same set as in (Turian et al., 2010b)

4.3 MWE Identification

Training set: randomly sampled 500 documents from Nathanas corpus.
Validation set: randomly sampled 100 documents from Nathanas corpus.
Test set: remaining 123 documents from Nathanas corpus..
Feature space: the same set as in (Schneider et al., 2014)

4.4 Named entity recognition

Training set: CoNLL03 train set.
Validation set: CoNLL03 development set.
Test set: CoNLL03 test set and MUC7. We report model performances on these two test sets respectively.
Feature space: the same set as in (Turian et al., 2010b)

5 Experimental Results and Discussion

Acknowledgments

Anonymised
Anonymised
Anonymised
Anonymised
Anonymised
Anonymised

References

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13(1):281–305.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

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

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *HLT-NAACL*, pages 380–390.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*.

Nathan Schneider, Spencer Onuffer, Nora Kazour, Emily Danchik, Michael T. Mordowanec, Henrietta Conrad, and Noah A. Smith. 2014. Comprehensive annotation of multiword expressions in a social web corpus. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, pages 455–461, Reykjavík, Iceland, May. ELRA.

Joseph Turian, Dpartement Dinformatique Et, Recherche Oprationnelle (diro, Universit De Montral, Lev Ratinov, and Yoshua Bengio. 2010a. Word representations: A simple and general method for semisupervised learning. In *In ACL*, pages 384–394.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010b. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.