Evaluation of Word Embeddings for Sequence Tagging Tasks

A Anonymous **B Anonymous**

Abstract

XXX

1 Introduction

In the last years, distributed word representations have been applied to several NLP tasks. Inspired by distributional semantics models, distributed word representation methods represent each word as a continuous vector, where similar words have a similar vector representation, therefore, capturing the similarity between words.

The resulting vectors can be used as features in many NLP applications and it has been shown that they outperform methods that treats words as atomic units (). Their attractiveness relies in the ability to learn word representations in an unsupervised way, thus directly providing lexical features from big amounts of unlabelled data and, therefore, alleviating the cost of human annotations. It has been also claimed that word embeddings have the ability to connect out of vocabulary words to known ones. Hence, suggesting that word embeddings are a good resource for applications that need to be adapted to a certain domain, different from the one the application have been tuned for. For example,... Another property attribute to word embeddings is their capacity to encouraging common behaviour among related in-vocabulary words, for instance...

Short sentence about the architectures

As with other learning methods, it is well known that the performance of machine learning algorithms heavily depends on parameter optimization, the size of the training data used and the applications they target. For example, (Turian et al., 2010) shows that the optimal word embedding dimensions are task specific. Moreover, there are several word embeddings methods, which used different algorithms and resources. Some methods involve feedback from the end task when learning (or fine-tuning) the word representations and others do not. Learning algorithm that involves fine-tuning are supposed to perform better since word representations become task-specific, at the cost of performing worst for out of vocabulary words. But still, there is not systematic comparison between these two methods.

In this paper, we perform an extensive evaluation of five word embedding approaches under fixed experiment conditions, and evaluate them over different sequence labelling tasks: POS-tagging, chunking, NER and MWE (Multi Word Expression Identification), within the following aims: (i) perform a fair comparison of different word embeddings algorithms. This includes running different word embeddings algorithms under controlled conditions, for example, use the same training set, the same preprocessing, etc.; (ii) measure the influence of word embeddings in sequence labeling tasks in semi-supervised settings (fine-tuning); (iii) systematically compared the usefulness of word embedding versus unigram features for sequence tagging. (iv) use word embeddings for MWE. To the best of our knowledge, word embeddings have not been used for this task before;

2 Related Work

Word embedding learning methods are the new generation of distributional semantics models and have been applied to several NLP tasks that we summaries in this section.

Collobert et al. (2011) proposed a neuronal network architecture that learn word embeddings and use them in POS-tagging, chunking, NER and SRL. Without specializing their architecture for the mentioned tasks, they achieve close state-of-the-art performance. After including specialized features (e.g., word suffixes for POS-tagging; Gazzeters for NER, etc.) and other tricks like cascading and ensambling classifiers, achieve com-

petitive state-of-the-art performance. Similarly, Turian et al. (2010) explored the impact of using word features learned from cluster-based and word embeddings representations for NER and chunking. They conclude that unsupervised word representation improve NER and chunking, and that combining different word representations can further improve the performance. Word representation from Brown clusters have been also shown to enhance Twitter POS tagging Owoputi et al. (2013).

Schneider et al. (2014a) presented a MWE analyser that, among other features, used unsupervised word clusters. They observed that the clusters were useful for identifying words that usually belong to proper names, which are considered MWE in the data set used. Nevertheless, they mentioned that it is difficult to measure the impact of the word embeddings features, since other features may capture the same information.

Word embeddings have been also used as features for syntactic dependency parsing and constituent parsing. Bansal et al. (2014) used word embeddings as features for dependency parsing, which used the syntactic dependency context instead of the linear context in raw text. They found that simple attempts based on discretization of individual word vector dimensions do not improve parsing. Only when performing hierarchical clustering of the continuous word vectors then using features based on the hierarchy, they gain performance. They also pointed out that ensemble of different word embeddings representations improved performance. Within the same aim, (Andreas and Klein, 2014) explores the used of word embeddings for constituency parsing and conclude that the information they provide might be redundant with the one acquire by a syntactic parser trained with a small amount of data. Others that boost the performance when including word embeddings representations for syntactic parsing includes (Koo et al., 2008; Koo et al., 2010; Haffari et al., 2011; Tratz and Hovy, 2011).

Word embedding have also been applied to other (non-sequential NLP) tasks such as supersense tagging (Edouard Grave and Bach, 2013); grammar induction (Spitkovsky et al., 2011) and semantic task such as semantic relatedness, synonymy detection, concept categorization selection preferences and analogy (Baroni et al., 2014)

Glove parameters	**Value**
Size of word vectors	
Discard words below frequency	5
Number of training iterations	25
Max epochs	50
Initial learning rate	0.05

Table 1: Glove algorithm parameters

Skip-gram parameters	**Value**
Word vector size	
Context window size	
Learning algorithm	10 negative samples
Discard words below frequency	5
Max epochs	50
Initial learning rate	0.025

Table 2: Skip-gram algorithm parameters

3 Learning Word Representations

Distributed word representation methods represent each word as a continuous vector, where similar words have a similar vector representation, therefore, capturing the similarity between words.

Example

The estimation of the word vectors can be carry out with different models architectures. We evaluate five different word embeddings learning algorithms, which are the following:

- Glove (Pennington et al., 2014)
- Skip-gram (Mikolov et al., 2013)
- CBOW (Mikolov et al., 2013)
- Neural language model (Collobert et al., 2011)
- Brown cluster (Brown et al., 1992)

The first four methods were chosen because they are recent state-of-the-art word embedding methods and because their software is available. The final method (Brown clusters) was selected as a benchmark word representation, which makes use of hard word clusters rather than a distributed representation.

Discuss here about fundamental differences between the above methods

CBOW parameters	**Value**
Word vector size	
Context window size	
Discard words below frequency	5
Max epochs	50
Initial learning rate	0.025

Table 3: Skip-gram algorithm parameters

Brown cluster parameters	Value
Number of clusters	
Discard words below frequency	5

Table 4: Brown algorithm parameters

3.1 Datasets

It is well known that the choice of a corpora have an important effect in the final accuracy of machine learning algorithms. Therefore, each word embedding method was trained with the same corpora (Table 5). The main reason of choosing the corpora is that they are publicly available.

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

Table 5: Corpora used to learn word embeddings

3.2 Preprocessing

All text are preprocessed with Stanford sentence splitter and tokeniser. All consecutive digit substrings are replaced by NUMf, where f is the length of the digit substring. For example, "10.20" is replaced by "NUM2.NUM2".

4 Sequence Tagging Tasks

We evaluate different word representations in four different sequence tagging tasks: POS tagging, chunking, NER and MWE identification. For each task, we feed learned word representations into a first order linear-chain graph transformer (Collobert et al., 2011), and trained them by using the online learning algorithm AdaGrad (Duchi et al., 2011). If we do not update word representations during training, the graph transformer is equivalent to a linear-chain CRF. For each model taking distributed word representations as word features, we consider two settings:

- Graph transformer *does not* fine tune word representations during training.
- Graph transformer fine tunes the word representations during training.

For each task, we consider also CRF models with hand-crafted features, which use one-hot representation for each unigram.

For each task, we split the task specific corpus into a training set, validation set, and a test set (see Table 6). If a corpus already provides fixed splits, we reuse them. For POS tagging and NER, we also evaluate the models with a out-of-domain corpus, which has similar annotation schema as the respective training corpus.

In order to have fair and reproducible experimental results, we tuned the hyperparameters with random search (Bergstra and Bengio, 2012). We randomly sampled 50 distinct hyperparameter sets with the same random seed for the models that do not update word embeddings, and sampled 100 distinct hyperparameter sets for the models that update word embeddings. For each set of hyperparameters, we train a model on its training set and pick up the best one based on its performance on its validation set (Turian et al., 2010). Note that, we also consider word vector size and context window size of distributed word representation, and number of clusters of brown clustering as the hyperparameters. This is achieved by mapping each possible hyperparameter combination to the word representation files trained with these parameters. Their ranges are listed below.

- **Word vector size**: [25, 50, 100, 200].
- **Context window size**: [5, 10, 15].
- **Number of brown clusters**: [250, 500, 1000, 2000, 4000].

However, for the models that update word representations, we always found under-performed hyperparameters after trying out all hyperparameter combinations, because they have more hyperparameters than the models that do not update word representations. Then for each distributed word representations, we reuse all hyperparameters of the models that do not update word representations, only tune the hyperparameters of AdaGrad for the word representation layer. This method requires only 32 additional runs for each model updating embeddings and achieves consistently better results than 100 random draws.

The final evaluation is carried out in a semi-supervised setting. We split the training set into 10 partitions at log scale. That means, the second smallest partition will be twice the size of the smallest partition. We created 10 training sets with incremental size by merging these partitions from the smallest one to the largest one, and evaluate them all on the same designated test sets.

We adopt the most commonly used F1 measures as the evaluation metric for all tasks except POS tagging, for which we use per-word accuracy. In order to evaluate model performance on unknown words, we report also the accuracy for the words that do not occur in the training set.

In addition, we also set up experiments to verify if CRF/graph transformer requires different feature design for different kinds of pre-trained word embeddings. This is achieved by adding a hidden layer between CRF and distributed word representations. For each context word, the hidden

	Training	Validation	Test	Feature space
POS-Tagging	0-18 WSJ	19-21 WSJ	22-24 of WSJ; English Web-Treebank	as in (Collobert et al., 2011)
Chunking	WSJ	1000 sentences WSJ	CoNLL-2000	as (Turian et al., 2010)
NER	CoNLL-2003 train set	CoNLL-2003 dev. set	CoNLL-2003 test set; MUC7	as in (Turian et al., 2010)
MWE	500 documents from	100 documents from	123 documents	as in (Schneider et al., 2014b)

Table 6: Datasets and features for each task.

Task	Benchmark	Us
POS-Tagging	(Accuracy) 97.24 (**?**)	
Chunking	(F1) 94.29 (**?**)	
NER	(F1) 89.31 (**?**)	
MWE	(F1) 57.71 (**?**)	

layer computes the element-wise multiplication of its embedding with the embedding of the current word embedding, and the representation of current word stays the same. The results of this approach are not plotted because this method leads only to marginal improvement.

5 Experimental Results and Discussion

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?

6 Conclusion

Acknowledgments

Anonymised
Anonymised
Anonymised
Anonymised
Anonymised
Anonymised

References

Jacob Andreas and Dan Klein. 2014. How much do word embeddings encode about syntax? In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 822–827, Baltimore, USA.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 809–815, Baltimore, USA.

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, Maryland, June. Association for Computational Linguistics.

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

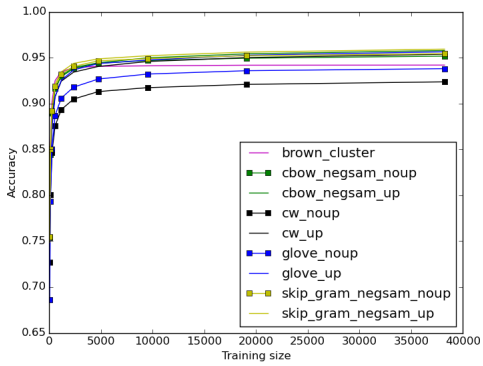
Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479.

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.

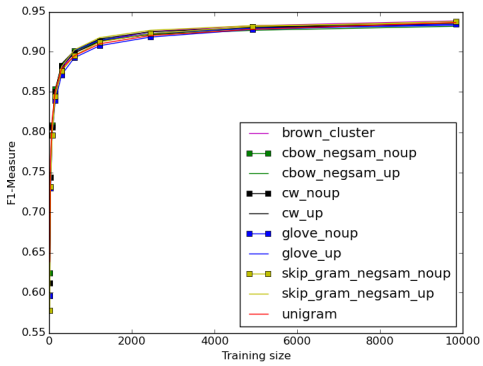
John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159.

Guillaume Obozinski Edouard Grave and Francis Bach. 2013. Hidden markov tree models for semantic class induction. In *Proceedings of CoNLL*.

Figure 1: Best results for each method for POS-Tagging and Chunking

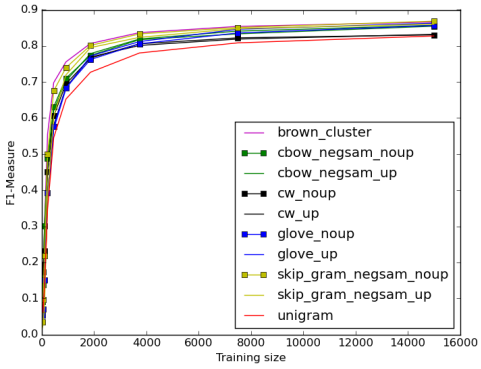


(a) POS-Tagging results

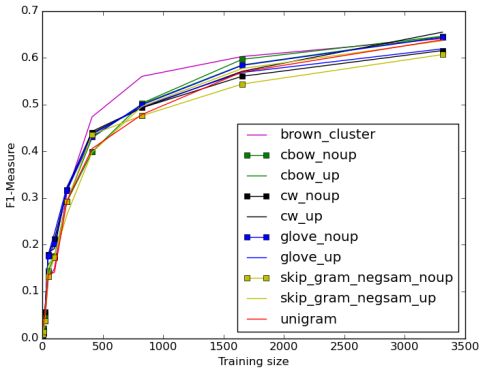


(b) Chunking results

Figure 2: Best results for each method for NER and MWE

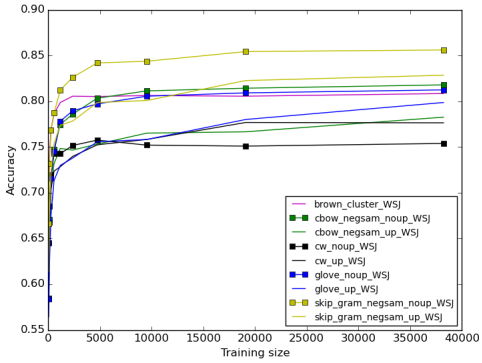


(a) NER results

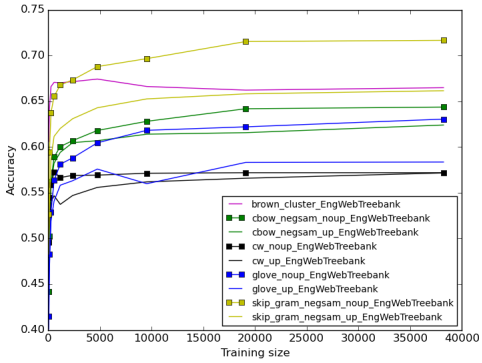


(b) MWE results

Figure 3: POS-Tagging out-of-vocabulary-words accuracy for *in-domain* and *out-of-domain* test sets

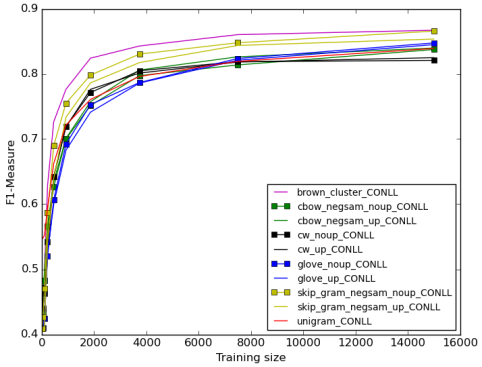


(a) *in domain*

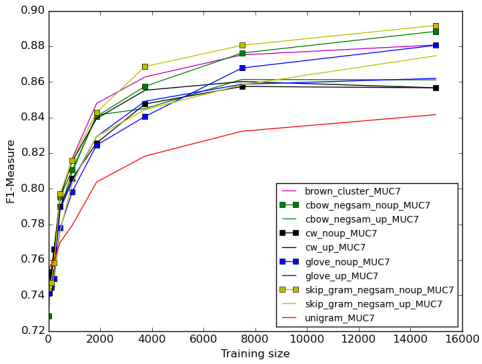


(b) *out-of-domain*

Figure 4: NER out-of-vocabulary-words accuracy for *in-domain* and *out-of-domain* test sets

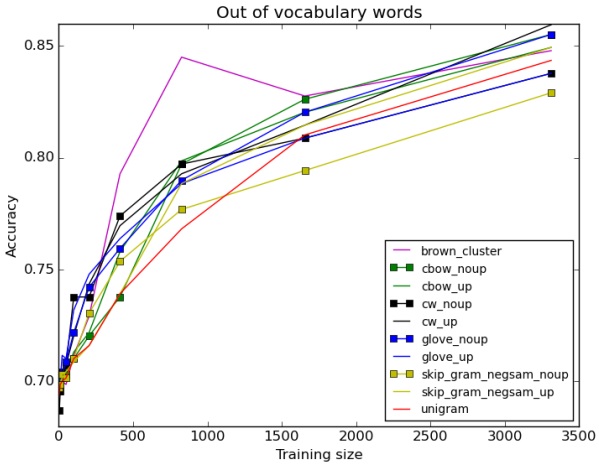


(a) *in-domain*



(b) *out-of-domain*

Figure 5: MWE out-of-vocabulary-words accuracy for *in-domain* test set



Gholamreza Haffari, Marzieh Razavi, and Anoop Sarkar. 2011. An ensemble model that combines syntactic and semantic clustering for discriminative dependency parsing. In *ACL (Short Papers)*, pages 710–714. The Association for Computer Linguistics.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *In Proc. ACL/HLT*.

Terry Koo, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1288–1298, Stroudsburg, PA, USA. Association for Computational Linguistics.

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, Emily Danchik, Chris Dyer, and Noah A. Smith. 2014a. Discriminative lexical semantic segmentation with gaps: Running the mwe gamut. *Transactions of the Association of Computational Linguistics*, 2(1):193–206.

Nathan Schneider, Spencer Onuffer, Nora Kazour, Emily Danchik, Michael T. Mordowanec, Henrietta Conrad, and Noah A. Smith. 2014b. 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.

Valentin I. Spitkovsky, Hiyan Alshawi, Angel X. Chang, and Daniel Jurafsky. 2011. Unsupervised dependency parsing without gold part-of-speech tags. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1281–1290, Stroudsburg, PA, USA. Association for Computational Linguistics.

Stephen Tratz and Eduard Hovy. 2011. A fast, accurate, non-projective, semantically-enriched parser. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1257–1268, Stroudsburg, PA, USA. Association for Computational Linguistics.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. 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.