Evaluation of Word Embeddings for Sequence Tagging Tasks

A Anonymous **B Anonymous**

Abstract

XXX

1 Introduction

Tim

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...

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

Gabriela

Word embedding learning methods 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 competitive 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)

3 Self-taught Learning for Sequence Tagging

This section might go to introduction The idea of learning word representations for downstream NLP applications embraces the idea of self-taught learning (). Self-taught learning starts with learning initial data representations from a large amount of unlabelled data, which may not be directly relevant to target applications. The learned representations are then fed as features into models of target applications, and may be fine-tuned during training to adapt to the target needs. Learning from a random sample of unlabelled data is distinct from semi-supervised learning (), which makes an assumption that the unlabelled data shares the same distribution as the labelled training data. In the following, we will introduce the recent advances of learning word representations and apply them for a range of sequence tagging tasks by using graph transformers ().

4 Word Representations

The distributional hypothesis in linguistics suggests that "a word is characterised by the company it keeps" (**?**). Words that are used in the similar contexts tend to have similar semantic and syntactic properties. Capturing distributional similarity is the underlying idea of all word representation learning methods.

4.1 Types of Word Representations

Based on the ways of constructing word representations, Turian et al. (2010) categorise these methods into three types: *Distributinoal representation*, *Cluster-based representation*, and *Distributed representation*.

Distributional representation methods map each word w to its context word vector $\mathbf{C}_w$, which is built based on coocurrence counts between w and the words surrounding it. The learning methods store either directly the cooccurence count between two words w and i in C_{wi} () or project the concurrence counts between words into a lower dimensional space by using techniques such as SVD () and LDA ().

The methods of *Cluster-based representation* build clusters of words by applying either soft- or hard clustering algorithms. Some of them also rely on cooccurence matrix of words (). The Brown algorithm () is the most famous one in this category.

A *distributed representation* takes the form of a dense, low-dimensional, and continuous-valued vector. It is compact and stores latent features of a word. This kind of representations are learned with various neural language models () in the hope of capturing both syntactic and semantic properties of words.

4.2 Selected Word Representations

For the sequence tagging tasks, we choose five top performed word representations in various empirical studies as candidates : Brown clustering (), CW (), Skip-gram (), CBOW (), and Glove (). The key idea of all these word models is to estimate the probability of word sequences. Formally, given a word sequence $\mathbf{w} =< w_1, w_2, ..., w_T >$, the training objective of these models is to maximise the log probability.

$$p(\mathbf{w}) = \frac{1}{T}\sum_{k=1}^{T} log(p(w_k|w_{j \in c_{k-m}^{k+n}})) \qquad (1)$$

where c_{k-m}^{k+n} denotes a sub-sequence $< w_{k-m}, ..., w_{k-1}, w_k, w_{k+1}, ..., w_{k+n} >$ of length $n - m - 1$, which is the local context of w_k. All models except CW choose $n = -1$ and $m > 0$ and exclude the word w_k from the local context.

The key differences among these models are the parameterisation of the factors $p(w_k|w_{j \in c_{k-m}^{k+n}})$ as well as the training loss functions. Brown clustering introduces a finite set of word classes V for each word,

4.3 Building Word Representations

For a fair comparison, we train each kind of word embedding on a combination of all corpora in Table 1. The joint corpus was preprocessed with the Stanford sentence splitter and tokenizer. All consecutive digit substrings were replaced by NUM*f*, where *f* is the length of the digit substring (e.g., “10.20” is replaced by “NUM2.NUM2”.

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

Table 1: Corpora used to learn word embeddings

5 Sequence Tagging Tasks

Lizhen

We evaluate different word representations in four different sequence tagging tasks: POS tagging, chunking, NER and MWE identification.

For each sequence tagging 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). For each model taking distributed word representations as word features, we consider two settings:

- Graph transformer *does not* fine tune word representations during training (this is equivalent to a linear-chain CRF);
- Graph transformer fine tunes the word representations during training.

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

We split the task specific corpus into a training set, validation set, and a test set (see Table 2). If a corpus already provides fixed splits, we reuse them. For POS-tagging and NER, we also evaluated the models with a out-of-domain corpus (English Web-Treebank and MUC-7, respectively), 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].

	Training set	Validation set	Test set	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 in (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 2: Datasets splits and feature space for each sequence tagging task.

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 each of them on the same designated test sets.

We adopted the most commonly used F1 measure as the evaluation metric for all tasks except POS tagging, for which we use per-word accuracy. In order to evaluate model performance on out-of-vocabulary (unknown) words, we reported 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 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.

6 Experimental Results and Discussion

Lizhen, Gabriela and Nathan?

As a reference, we compared our best results for each task with their corresponding benchmarks. For all the task, we reach a comparable performance to the state-of-the-art methods, except for NER that we are 2.5 points below, because... (Table3). However, in this paper, we do not aim to maximize the absolute performance of the tasks under study, but rather to study the impact of word embeddings for sequence tagging tasks under control settings.

Figure 1 and Figure 2 summarized the results obtained for each task and each word embedding method in heat-maps plots. Across the different tasks, we could not find any trend that tell which word embedding methods is the best and under which settings, since the difference between them are not significant (see that all methods are in green when all the available training data was used).

We observed also that word embeddings are especially helping POS-tagging and chunking when there are only several hundreds of training instances. These early improvements are less evident for NER and MWE. We attribute this to: i) NER and MWE are more difficult tasks than POS-tagging and chunking; ii) NER and MWE require more training data; iii) the performance of NER and MWE heavily dependent on complex features such as gazetteers and lexicons like Wordnet, which are not captured by the feature representation learned from unlabelled data, meanwhile the standard features used in POS-tagging and chunking (e.g., stemming) are more likely to be capture by the word embeddings, because...

As expected, word embeddings and Brown clustering excel in out-of-domain performance.

Fine-tuning can correct poorly learned word representations but can be overfitted if unsupervisely learned ones are already good.

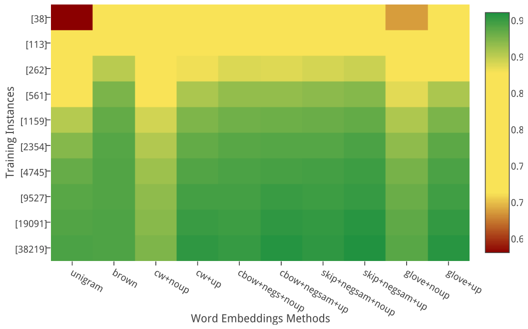
Because we can either update pre-trained word embeddings during training or not, through the evaluation, we want to answer the following questions:

Table 3: Benchmark results vs. our best results

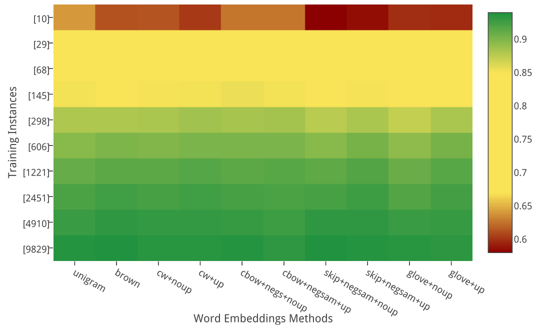
Task	**Benchmark**	**Us**
POS-Tagging	(Accuracy) 97.24 (Toutanova et al., 2003)	0.9592 (skip-gram negsam+up)
Chunking	(F1) 0.9429 (Sha and Pereira, 2003)	0.9386 (Brown cluster v2000+)
NER	(F1) 0.8931 (Ando and Zhang, 2005)	0.8686 (skip-gram negsam+noup)
MWE	(F1) 0.6253 (Schneider et al., 2014a)	0.6546 (cw+up)

- 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?

Figure 1: Best results for each method for POS-Tagging and Chunking. The x-axis correspond to the different word embeddings methods and the y-axis to the 10 training partitions at log scale. Green color stand for high performance, while red color stands for low performance.

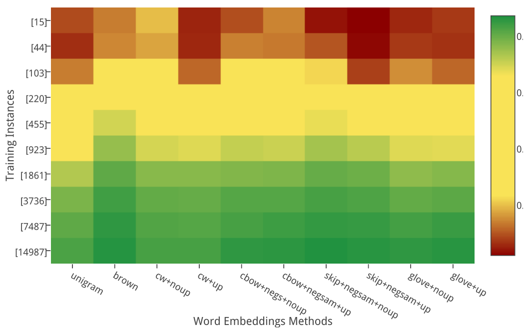


(a) POS-Tagging Accuracy

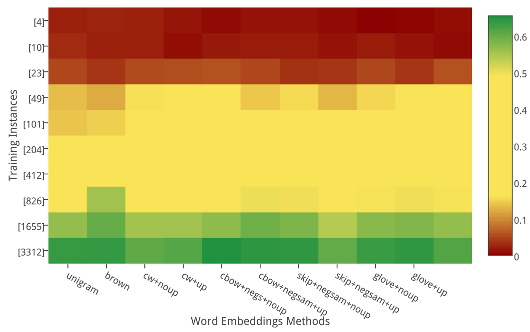


(b) Chunking F1-Measure

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



(a) NER F1-Measure



(b) MWEs F1-Measure

6.1 Result tables

The first column of each table contains the number of training sentences.

6.1.1 Best hyperparameters for All Tasks

6.1.2 Out of Vocabulary results for All Tasks

6.1.3 Out of domain Results for NER and POS

7 Conclusion

Acknowledgments

Anonymised
Anonymised
Anonymised
Anonymised
Anonymised
Anonymised

References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.*, 6:1817–1853, December.

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*.

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.

Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 134–141, Stroudsburg, PA, USA. Association for Computational Linguistics.

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.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, 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.

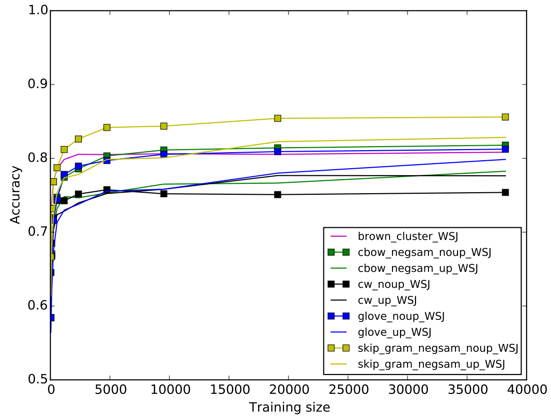


Figure 3: POS-Tagging accuracy for *in domain* OOV

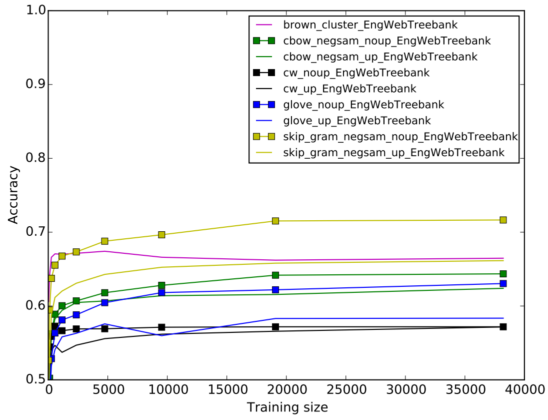


Figure 4: POS-Tagging accuracy for *out domain* OOV

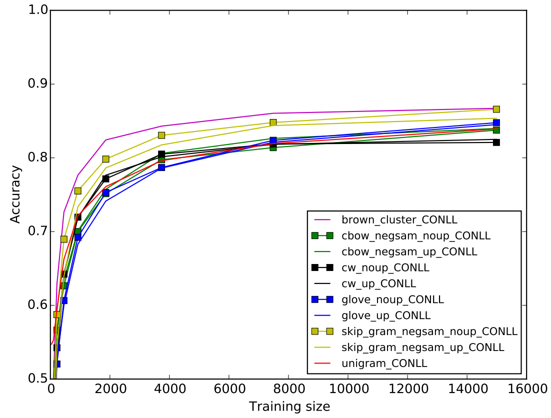


Figure 5: NER accuracy for *out domain* OOV

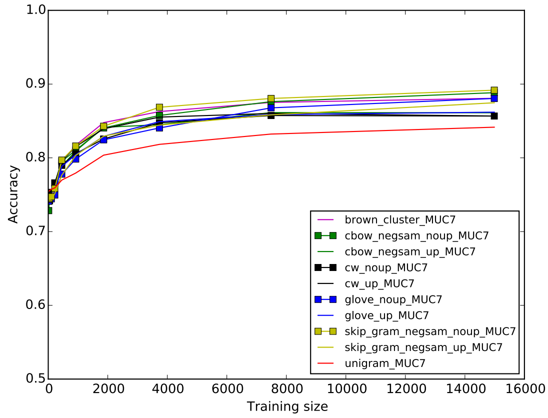


Figure 6: NER accuracy for *out domain* OOV

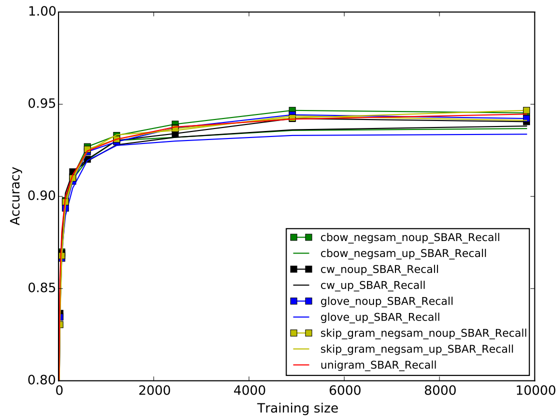


Figure 7: Chunking accuracy for OOV *in domain* OOV

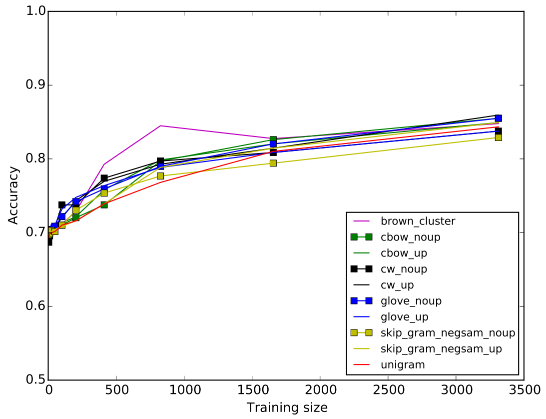


Figure 8: MWE accuracy for *out domain* OOV

label	unigram	brown+cluster+v4000	cw+noupdated	cw+updated	cbow+negsam+noupdated	cbow+negsam+updated	skip+gram+negsam+noupdated	skip+gram+negsam+updated	glove+noupdated	glove+updated
38	0.6293	0.7687	0.7267	0.742	0.7543	0.7457	0.7547	0.7486	0.6861	0.7172
113	0.7387	0.8527	0.8009	0.8266	0.8446	0.8386	0.8501	0.8482	0.793	0.8178
262	0.8208	0.9019	0.8468	0.8818	0.8896	0.8877	0.8921	0.8953	0.8503	0.8737
561	0.8721	0.9249	0.8755	0.9076	0.9162	0.9159	0.9185	0.921	0.8869	0.9071
1,159	0.9039	0.9346	0.8933	0.9243	0.9293	0.9303	0.9321	0.9344	0.9059	0.9253
2,354	0.9215	0.9391	0.9048	0.9341	0.938	0.9392	0.9403	0.9437	0.9178	0.9366
4,745	0.933	0.9405	0.9129	0.9404	0.9439	0.9444	0.9465	0.9486	0.9266	0.9435
9,527	0.9382	0.941	0.9173	0.9458	0.9468	0.9498	0.9489	0.952	0.9321	0.9474
19,091	0.9413	0.9417	0.9208	0.95	0.9494	0.9541	0.9521	0.956	0.9357	0.9526
38,219	0.9432	0.9419	0.9236	0.9537	0.9516	0.9577	0.9543	0.9592	0.9379	0.9563

Table 4: Accuracy of POS tagging evaluated on WSJ test set

label	unigram	brown+cluster+v4000	cw+noupdated	cw+updated	cbow+negsam+noupdated	cbow+negsam+updated	skip+gram+negsam+noupdated	skip+gram+negsam+updated	glove+noupdated	glove+updated
15	0.0873	0.1345	0.1933	0.0524	0.0917	0.1424	0.0339	0.0173	0.0543	0.0707
44	0.0618	0.1441	0.1725	0.0529	0.1378	0.13	0.0972	0.0256	0.0702	0.0629
103	0.1332	0.3023	0.2318	0.1159	0.3003	0.2955	0.2184	0.0774	0.1508	0.1144
220	0.3469	0.5553	0.45	0.3909	0.4886	0.4751	0.4993	0.3852	0.3924	0.3749
455	0.5457	0.6983	0.606	0.58	0.6311	0.6214	0.6752	0.6037	0.5764	0.5694
923	0.6526	0.7557	0.6949	0.6848	0.7104	0.7033	0.7396	0.7198	0.6842	0.6813
1,861	0.7271	0.8068	0.7666	0.7691	0.7736	0.7786	0.8013	0.7954	0.7622	0.7722
3,736	0.7804	0.8369	0.8059	0.8013	0.8174	0.8192	0.8333	0.8237	0.8059	0.8121
7,487	0.8084	0.854	0.8224	0.819	0.8333	0.841	0.851	0.8481	0.8359	0.8462
14,987	0.8276	0.8663	0.8306	0.832	0.8547	0.8573	0.8686	0.8603	0.8565	0.8631

Table 5: F1 Measure of NER evaluated on CoNLL test set

label	unigram	brown+cluster+v2000	cw+noupdated	cw+updated	cbow+negsam+noupdated	cbow+negsam+updated	skip+gram+negsam+noupdated	skip+gram+negsam+updated	glove+noupdated	glove+updated
10	0.6386	0.6108	0.6123	0.6006	0.6247	0.6247	0.5778	0.5832	0.5963	0.5952
29	0.764	0.7424	0.7442	0.7376	0.744	0.7375	0.7325	0.7317	0.7306	0.726
68	0.8148	0.8001	0.8065	0.8051	0.8088	0.8061	0.7961	0.8012	0.7961	0.7971
145	0.8519	0.8492	0.8506	0.851	0.8538	0.8518	0.8442	0.8496	0.8392	0.8459
298	0.8791	0.879	0.8818	0.885	0.8826	0.8834	0.876	0.8818	0.8715	0.8806
606	0.8951	0.8992	0.8987	0.9015	0.9009	0.901	0.8955	0.9024	0.8927	0.9019
1,221	0.9101	0.9139	0.913	0.9161	0.9134	0.9151	0.9125	0.9178	0.9077	0.9145
2,451	0.9203	0.9253	0.9221	0.9251	0.9221	0.9214	0.9227	0.9269	0.9184	0.9231
4,910	0.9281	0.9326	0.931	0.9303	0.9292	0.9268	0.9316	0.9326	0.9279	0.9294
9,829	0.9365	0.9386	0.935	0.9348	0.9362	0.9319	0.9378	0.9359	0.9347	0.933

Table 6: F1 Measure of chunking evaluated on CoNLL test set

label	unigram	brown+cluster+v4000	brown+cluster+nathn	cw+noupdated	cw+updated	cbow+noupdated	cbow+updated	skipgram+noupdated	skipgram+updated	glove+noupdated	glove+updated
4	0.0241	0.0255	0.0232	0.0146	$8.2 \cdot 10^{-3}$	0.0122	0.0138	$8.4 \cdot 10^{-3}$	0	$3.4 \cdot 10^{-3}$	0.0111
10	0.0306	0.024	0.0239	0.0103	0.0172	0.0205	0.0206	0.0137	0.0206	0.0138	$6.8 \cdot 10^{-3}$
23	0.0505	0.0375	0.054	0.0564	0.0603	0.0505	0.0373	0.0374	0.0507	0.0374	0.0599
49	0.1354	0.1241	0.1623	0.1782	0.1825	0.1437	0.1573	0.1314	0.1544	0.1751	0.1782
101	0.1412	0.1494	0.1785	0.2121	0.192	0.1752	0.173	0.1723	0.1743	0.2021	0.2245
204	0.2933	0.311	0.3036	0.3173	0.3204	0.2921	0.2926	0.2926	0.2644	0.3156	0.3234
412	0.4054	0.4733	0.4658	0.439	0.4424	0.3985	0.4287	0.4358	0.3975	0.4312	0.4363
826	0.4785	0.5601	0.4929	0.493	0.493	0.5027	0.5011	0.4762	0.4959	0.4994	0.4955
1,655	0.5695	0.6025	0.5578	0.5602	0.5709	0.5965	0.5845	0.5437	0.5768	0.5836	0.5684
3,312	0.6384	0.6409	0.6077	0.6153	0.6546	0.6459	0.645	0.6067	0.6363	0.6434	0.619

Table 7: F1 Measure of MWE identification evaluated on MWE test set

label	unigram	brown+cluster+v4000	cw+noupdated	cw+updated	cbow+negsam+noupdated	cbow+negsam+updated	skip+gram+negsam+noupdated	skip+gram+negsam+updated	glove+noupdated	glove+updated
38	0.4546	0.7018	0.6455	0.6302	0.5844	0.5731	0.6665	0.6135	0.584	0.5644
113	0.525	0.7404	0.6851	0.6676	0.6705	0.6524	0.732	0.6782	0.6676	0.6215
262	0.6306	0.7691	0.7196	0.7036	0.7164	0.7047	0.7684	0.7229	0.716	0.6756
561	0.6815	0.7862	0.7429	0.7236	0.7473	0.732	0.7873	0.7538	0.7447	0.7135
1,159	0.717	0.7985	0.7425	0.7284	0.7745	0.7484	0.812	0.7738	0.7782	0.7298
2,354	0.7377	0.8055	0.7516	0.7396	0.7858	0.7465	0.8262	0.7782	0.7895	0.7378
4,745	0.7503	0.8051	0.7575	0.7524	0.8033	0.7531	0.8418	0.7985	0.7971	0.7556
9,527	0.7602	0.8062	0.752	0.7582	0.8113	0.7651	0.8436	0.8007	0.8055	0.7582
19,091	0.7673	0.8055	0.7509	0.7767	0.8142	0.7665	0.8542	0.8225	0.8091	0.78
38,219	0.7719	0.8084	0.7538	0.7764	0.8178	0.7825	0.856	0.8284	0.8124	0.7985

Table 8: Accuracy of POS tagging evaluated on out-of-vocabulary words in WSJ test set

label	unigram	brown+cluster+v4000	cw+noupdated	cw+updated	cbow+negsam+noupdated	cbow+negsam+updated	skip+gram+negsam+noupdated	skip+gram+negsam+updated	glove+noupdated	glove+updated
15	0.5468	0.4234	0.4728	0.4281	0.4351	0.4523	0.4091	0.4079	0.4102	0.4136
44	0.5478	0.4281	0.4406	0.4047	0.4351	0.4481	0.4263	0.4047	0.4099	0.411
103	0.5528	0.4986	0.4632	0.4221	0.4825	0.4994	0.47	0.4107	0.4245	0.4216
220	0.5948	0.6291	0.5425	0.5277	0.5664	0.5672	0.5875	0.542	0.5204	0.5266
455	0.6625	0.7265	0.6423	0.6423	0.6265	0.6345	0.6896	0.6395	0.6064	0.5999
923	0.7215	0.7764	0.7195	0.7187	0.7	0.7011	0.7551	0.7336	0.6922	0.6829
1,861	0.7609	0.8243	0.7715	0.7764	0.7515	0.7575	0.7983	0.7863	0.7531	0.7414
3,736	0.7962	0.843	0.805	0.8014	0.7975	0.8061	0.8305	0.8175	0.7868	0.7861
7,487	0.8194	0.8604	0.8188	0.8183	0.8139	0.8264	0.8479	0.8438	0.8238	0.8212
14,987	0.8392	0.8672	0.8209	0.8251	0.8375	0.8399	0.8659	0.8537	0.8477	0.8451

Table 9: Accuracy of NER evaluated on out-of-vocabulary words in CoNLL test set

label	unigram	brown+cluster+v2000	cw+noupdated	cw+updated	cbow+negsam+noupdated	cbow+negsam+updated	skip+gram+negsam+noupdated	skip+gram+negsam+updated	glove+noupdated	glove+updated
10	0.7958	0.7907	0.7849	0.7988	0.7751	0.7788	0.7744	0.774	0.7737	0.7717
29	0.8548	0.8447	0.8366	0.8468	0.8349	0.8393	0.8304	0.8366	0.8345	0.8383
68	0.8808	0.8719	0.8692	0.8733	0.8699	0.8753	0.8678	0.8675	0.8665	0.8702
145	0.8998	0.9001	0.8967	0.9018	0.8977	0.8974	0.897	0.895	0.8936	0.8896
298	0.9116	0.9096	0.9134	0.913	0.9106	0.9093	0.91	0.9093	0.9083	0.9045
606	0.9247	0.9212	0.9201	0.9191	0.9269	0.9198	0.9256	0.9246	0.9242	0.9195
1,221	0.931	0.931	0.9303	0.928	0.9331	0.9303	0.931	0.9334	0.9297	0.9276
2,451	0.9377	0.9354	0.9341	0.932	0.9392	0.932	0.9368	0.9358	0.9371	0.93
4,910	0.9419	0.9405	0.9422	0.9361	0.9467	0.9358	0.9426	0.9436	0.9443	0.9331
9,829	0.9446	0.9439	0.9405	0.9382	0.9453	0.9368	0.9467	0.9412	0.9422	0.9337

Table 10: Accuracy of Chunking evaluated on out-of-vocabulary words in CoNLL test set

label	unigram	brown+cluster+v4000	cw+noupdated	cw+updated	cbow+noupdated	cbow+updated	skipgram+noupdated	skipgram+updated	glove+noupdated	glove+updated
4	0.6942	0.6971	0.687	0.6971	0.6957	0.6971	0.6971	0.6971	0.6971	0.6986
10	0.6986	0.6986	0.6957	0.6986	0.7	0.7	0.6986	0.6986	0.6986	0.7
23	0.7	0.7	0.7043	0.7043	0.7014	0.7	0.7029	0.7014	0.7029	0.7116
49	0.7014	0.6986	0.7072	0.7072	0.7029	0.7043	0.7014	0.7043	0.7087	0.7101
101	0.7101	0.7116	0.7377	0.7203	0.7101	0.713	0.7101	0.7087	0.7217	0.7319
204	0.7159	0.729	0.7377	0.7435	0.7203	0.7217	0.7304	0.7159	0.742	0.7478
412	0.7391	0.7928	0.7739	0.7696	0.7377	0.758	0.7536	0.7391	0.7594	0.7638
826	0.7681	0.8449	0.7971	0.7928	0.7971	0.7986	0.7768	0.7884	0.7899	0.7884
1,655	0.8101	0.8275	0.8087	0.8145	0.8261	0.8203	0.7942	0.8145	0.8203	0.8087
3,312	0.8435	0.8478	0.8377	0.8594	0.8551	0.8493	0.829	0.8493	0.8551	0.8377

Table 11: Accuracy of MWE identification evaluated on out-of-vocabulary words in MWE test set

label	unigram	brown+cluster+v4000	cw+noupdated	cw+updated	cbow+negsam+noupdated	cbow+negsam+updated	skip+gram+negsam+noupdated	skip+gram+negsam+updated	glove+noupdated	glove+updated
38	0.5345	0.7106	0.6303	0.636	0.6854	0.6681	0.6786	0.6631	0.5931	0.6249
113	0.6448	0.7898	0.7151	0.7468	0.779	0.7632	0.7819	0.7781	0.7023	0.7407
262	0.7279	0.8537	0.7704	0.8219	0.8407	0.8341	0.8389	0.8433	0.7712	0.8148
561	0.7967	0.8772	0.8067	0.8481	0.8686	0.8624	0.8659	0.8675	0.816	0.8464
1,159	0.8318	0.8879	0.8303	0.8633	0.882	0.8782	0.8856	0.882	0.8426	0.865
2,354	0.8511	0.8916	0.8464	0.8728	0.8895	0.8855	0.8938	0.8891	0.8575	0.8766
4,745	0.8656	0.8942	0.8563	0.8825	0.8965	0.8908	0.901	0.8971	0.8701	0.886
9,527	0.873	0.8926	0.8614	0.8877	0.8994	0.8961	0.9054	0.9017	0.876	0.8912
19,091	0.877	0.8908	0.8656	0.8913	0.9029	0.9003	0.9084	0.9044	0.8804	0.8979
38,219	0.8796	0.8911	0.8681	0.8959	0.9045	0.902	0.9105	0.9064	0.8836	0.9009

Table 12: Accuracy of POS tagging evaluated on English web treebank.

label	unigram	brown+cluster+v4000	cw+noupdated	cw+updated	cbow+negsam+noupdated	cbow+negsam+updated	skip+gram+negsam+noupdated	skip+gram+negsam+updated	glove+noupdated	glove+updated
15	0.0424	0.0589	0.0679	0.0451	0.0562	0.0785	0.0223	0.023	0.0306	0.0296
44	0.0487	0.0624	0.0615	0.0465	0.0608	0.0609	0.032	0.0304	0.0494	0.0568
103	0.0503	0.0837	0.0825	0.0687	0.0996	0.123	0.0479	0.0395	0.0553	0.0539
220	0.1054	0.227	0.1925	0.1788	0.2089	0.2315	0.1423	0.1118	0.1311	0.1525
455	0.3051	0.434	0.3769	0.3787	0.4156	0.4262	0.3976	0.3621	0.34	0.3903
923	0.3955	0.5136	0.4403	0.468	0.4897	0.5024	0.4881	0.461	0.445	0.4708
1,861	0.48	0.5834	0.52	0.5346	0.5629	0.5666	0.5784	0.5464	0.5294	0.5342
3,736	0.5622	0.6465	0.6034	0.6201	0.6368	0.6118	0.6663	0.6114	0.6109	0.6155
7,487	0.5956	0.6726	0.6334	0.6425	0.6765	0.6529	0.6794	0.65	0.6602	0.6491
14,987	0.6288	0.7012	0.6391	0.6521	0.7148	0.6838	0.7364	0.6964	0.6929	0.6754

Table 13: F1 measure of NER evaluated on MUC7 test set