Big Data Small Data, In Domain Out-of Domain, Known Word Unknown Word: The Impact of Word Representations on Sequence Labelling Tasks

Lizhen Qu[1,2], **Gabriela Ferraro**[1,2], **Liyuan Zhou**[1], **Weiwei Hou**[1],
Nathan Schneider[3] and **Timothy Baldwin**[1,4]

[1] NICTA, Australia
[2] The Australian National University
[3] University of Edinburgh
[4] The University of Melbourne

{lizhen.qu,gabriela.ferraro,liyuan.zho,weiwei.hou}@nicta.com.au
nschneid@cs.cmu.edu
tb@ldwin.net

Abstract

Word embeddings — distributed word representations that can be learned from unlabelled data — have been shown to have high utility in many natural language processing applications. In this paper, we perform an extrinsic evaluation of four popular word embedding methods in the context of four sequence labelling tasks: part-of-speech tagging, syntactic chunking, named entity recognition, and multiword expression identification. A particular focus of the paper is analysing the effects of task-based updating of word representations. We show that when using word embeddings as features, as few as several hundred training instances are sufficient to achieve competitive results, and that word embeddings lead to improvements over out-of-vocabulary words and also out of domain. Perhaps more surprisingly, our results indicate there is little difference between the different word embedding methods, and that simple Brown clusters are often competitive with word embeddings across all tasks we consider.

1 Introduction

Recently, distributed word representations have grown to become a mainstay of natural language processing (NLP), and have been shown to have empirical utility in a myriad of tasks (Collobert and Weston, 2008; Turian et al., 2010; Baroni et al., 2014; Andreas and Klein, 2014). The underlying idea behind distributed word representations is simple: to map each word w in vocabulary V onto a continuous-valued vector of dimensionality $d \ll |V|$. Words that are similar (e.g., with respect to syntax or lexical semantics) will ideally be mapped to similar regions of the vector space, implicitly supporting both generalisation across in-vocabulary (IV) items, and countering the effects of data sparsity for low-frequency and out-of-vocabulary (OOV) items.

Without some means of automatically deriving the vector representations without reliance on labelled data, however, word embeddings would have little practical utility. Fortunately, it has been shown that they can be "pre-trained" from unlabelled text data using various algorithms to model the distributional hypothesis (i.e., that words which occur in similar contexts tend to be semantically similar). Pre-training methods have been refined considerably in recent years, and scaled up to increasingly large corpora.

As with other machine learning methods, it is well known that the quality of the pre-trained word embeddings depends heavily on factors including parameter optimisation, the size of the training data, and the fit with the target application. For example, Turian et al. (2010) showed that the optimal dimensionality for word embeddings is task-specific. One factor which has received relatively little attention in NLP is the effect of "updating" the pre-trained word embeddings as part of the task-specific training, based on self-taught learning (Raina et al., 2007). Updating leads to word representations that are task-specific, but often at the cost of over-fitting low-frequency and OOV words.

In this paper, we perform an extensive evaluation of four recently proposed word embedding approaches under fixed experimental conditions, applied to four sequence labelling tasks: part-of-speech (POS) tagging, full-text chunking, named entity recognition (NER), and multiword expres-

Proceedings of the 19th Conference on Computational Language Learning, pages 83–93,
Beijing, China, July 30-31, 2015. ©2015 Association for Computational Linguistics

sion (MWE) identification.[1] We build on previous empirical studies (Collobert et al., 2011; Turian et al., 2010; Pennington et al., 2014) in considering a broader range of word embedding approaches and evaluating them over more sequence labelling tasks. In addition, we explore the following research questions:

RQ1: are word embeddings better than baseline approaches of one-hot unigram[2] features and Brown clusters?

RQ2: do word embeddings require less training data (i.e., generalise better) than one-hot unigram features? If so, to what degree can word embeddings reduce the amount of labelled data?

RQ3: what is the impact of updating word embeddings in sequence labelling tasks, both empirically over the target task and geometrically over the vectors?

RQ4: what is the impact of these word embeddings (with and without updating) on both OOV items (relative to the training data) and out-of-domain data?

RQ5: overall, are some word embeddings better than others in a sequence labelling context?

2 Word Representations

2.1 Types of Word Representations

Turian et al. (2010) identifies three varieties of word representations: *distributional*, *cluster-based*, and *distributed*.

Distributional representation methods map each word w to a context word vector $\mathbf{C}_w$, which is constructed directly from co-occurrence counts between w and its context words. The learning methods either store the co-occurrence counts between two words w and i directly in C_{wi} (Sahlgren, 2006; Turney and Pantel, 2010; Honkela, 1997) or project the concurrence counts between words into a lower dimensional space (Řehůřek and Sojka, 2010; Lund and Burgess, 1996), using dimensionality reduction techniques such as SVD (Dumais et al., 1988) or LDA (Blei et al., 2003).

Cluster-based representation methods build clusters of words by applying either soft or hard clustering algorithms (Lin and Wu, 2009; Li and McCallum, 2005). Some of them also rely on a co-occurrence matrix of words (Pereira et al., 1993). The Brown clustering algorithm (Brown et al., 1992) is the best-known method in this category.

Distributed representation methods usually map words into dense, low-dimensional, continuous-valued vectors, with $\mathbf{x} \in \mathbb{R}^d$, where d is referred to as the word dimension.

2.2 Selected Word Representations

Over a range of sequence labelling tasks, we evaluate four methods for inducing word representations: Brown clustering (Brown et al., 1992) ("BROWN"), the continuous bag-of-words model ("CBOW") (Mikolov et al., 2013a), the continuous skip-gram model ("SKIP-GRAM") (Mikolov et al., 2013b), and Global vectors ("GLOVE") (Pennington et al., 2014). All have been shown to be at or near state-of-the-art in recent empirical studies (Turian et al., 2010; Pennington et al., 2014).[3] The training of these word representations is unsupervised: the common underlying idea is to predict the occurrence of words in the neighbouring context. Their training objectives share the same form, which is a sum of local training factors $J(w, \text{ctx}(w))$,

$$L = \sum_{w \in T} J(w, \text{ctx}(w))$$

where T is the set of tokens in a given corpus, and $\text{ctx}(w)$ denotes the local context of word w. The local context of a word is conventionally its preceding m words, or alternatively the m words surrounding it. Local training factors are designed to capture the relationship between w and its local contexts of use, either by predicting w based on its local context, or using w to predict the context words. Other than BROWN, which utilises a cluster-based representation, all the other methods employ a distributed representation.

The starting point for CBOW and SKIP-GRAM is to employ softmax to predict word occurrence:

$$J(w, \text{ctx}(w)) = -\log \left(\frac{\exp(\mathbf{v}_w^{\mathrm{T}} \mathbf{v}_{\text{ctx}(w)})}{\sum_{j \in V} \exp(\mathbf{v}_j^{\mathrm{T}} \mathbf{v}_{\text{ctx}(w)})} \right)$$

[1] MWEs are lexicalized combinations of two or more simplex words that are exceptional enough to be considered as single units in the lexicon (Baldwin and Kim, 2010; Schneider et al., 2014a), e.g., *pick up* or *part of speech*.

[2] Word vectors with one-hot representation are binary vectors with a single dimension per word in the vocabulary (i.e., $d = |V|$), with the single dimension corresponding to the target word set to 1 and all other dimensions set to 0.

[3] The word embedding approach proposed in Collobert et al. (2011) is not considered because it was found to be inferior to our four target word embedding approaches in previous work.

where $\mathbf{v}_{\text{ctx}(w)}$ denotes the distributed representation of the local context of word w, and V is the vocabulary of a given corpus. CBOW derives $\mathbf{v}_{\text{ctx}(w)}$ based on averaging over the context words. That is, it estimates the probability of each w given its local context. In contrast, SKIP-GRAM applies softmax to each context word of a given occurrence of word w. In this case, $\mathbf{v}_{\text{ctx}(w)}$ corresponds to the representation of one of its context words. This model can be characterised as predicting context words based on w. In practice, softmax is too expensive to compute over large corpora, and thus Mikolov et al. (2013b) use hierarchical softmax and negative sampling to scale up the training.

GLOVE assumes the dot product of two word embeddings should be similar to the logarithm of the co-occurrence count X_{ij} of the two words. As such, the local factor $J(w, \text{ctx}(w))$ becomes:

$$g(X_{ij})(\mathbf{v}_i^{\text{T}}\mathbf{v}_j + b_i + b_j - \log(X_{ij}))^2$$

where b_i and b_j are the bias terms of words i and j, respectively, and $g(X_{ij})$ is a weighting function based on the co-occurrence count. This weighting function controls the degree of agreement between the parametric function $\mathbf{v}_i^{\text{T}}\mathbf{v}_j + b_i + b_j$ and $\log(X_{ij})$. Frequently co-occurring word pairs will have larger weight than infrequent pairs, up to a threshold.

BROWN partitions words into a finite set of word classes V. The conditional probability of seeing the next word is defined to be:

$$p(w_k|w_{k-m}^{k-1}) = p(w_k|h_k)p(h_k|h_{k-m}^{k-1})$$

where h_k denotes the word class of the word w_k, w_{k-m}^{k-1} are the previous m words, and h_{k-m}^{k-1} are their respective word classes. Then $J(w, \text{ctx}(w)) = -\log p(w_k|w_{k-m}^{k-1})$. Since there is no tractable method to find an optimal partition of word classes, the method uses only a bigram class model, and utilises hierarchical clustering as an approximation method to find a sufficiently good partition of words.

2.3 Building Word Representations

To ensure the comparison of different word representations is fair, we train BROWN, CBOW, SKIP-GRAM, and GLOVE on a fixed corpus, comprised of freely available corpora, as detailed in Table 1. The joint corpus was preprocessed with the Stanford CoreNLP sentence splitter and tokeniser. 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 (Han et al., 2013)	48.1GB	3G
One Billion (Chelba et al., 2013)	4.1GB	1G
English Wikipedia	49.6GB	3G

Table 1: Corpora used to pre-train the word embeddings

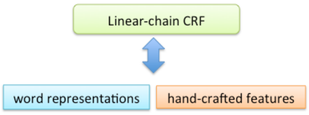


Figure 1: Linear-chain graph transformer

The dimensionality of the word embeddings and the size of the context window are the key hyperparameters when learning distributed representations. We use all combinations of the following values to train word embeddings on the combined corpus:

- **Embedding dim.** $d \in \{25, 50, 100, 200\}$
- **Context window size** $m \in \{1, 5, 10\}$

BROWN requires only the number of clusters as a hyperparameter. Here, we perform clustering with $b \in \{250, 500, 1000, 2000, 4000\}$ clusters.

3 Sequence Labelling Tasks

We evaluate the different word representations over four sequence labelling tasks: POS tagging (POS tagging), full-text chunking (Chunking), NER (NER), and MWE identification (MWE). For each task, we fed features into a first-order linear-chain graph transformer (Collobert et al., 2011) made up of two layers: the upper layer is identical to a linear-chain CRF (Lafferty et al., 2001), and the lower layer consists of word representation and hand-crafted features. If we treat word representations as fixed, the graph transformer is a simple linear-chain CRF. On the other hand, if we can treat the word representations as model parameters, the model is equivalent to a neural network with word embeddings as the input layer, as shown in Figure 1. We trained all models using AdaGrad (Duchi et al., 2011).

As in Turian et al. (2010), at each word position, we construct word representation features from the words in a context window of size two to either side of the target word, based on the pre-trained representation of each word type. For BROWN, the features are the prefix features extracted from word clusters in the same way as Turian et al. (2010). As a baseline (and to test **RQ1**), we include a one-hot representation (which is equivalent to a linear-chain CRF with only lexical context features).

Our hand-crafted features for POS tagging, Chunking and MWE, are those used by Collobert et al. (2011), Turian et al. (2010) and Schneider et al. (2014b), respectively. For NER, we use the same feature space as Turian et al. (2010), except for the previous two predictions, because we want to evaluate all word representations with the same type of model — a first-order graph transformer.

In training the distributed word representations, we consider two settings: (1) the word representations are fixed during sequence model training; and (2) the graph transformer updated the token-level word representations during training.

As outlined in Table 2, for each sequence labelling task, we experiment over the de facto corpus, based on pre-existing training–dev–test splits where available:[4]

POS tagging: the Wall Street Journal portion of the Penn Treebank (Marcus et al. (1993): "WSJ") with Penn POS tags

Chunking: the Wall Street Journal portion of the Penn Treebank ("WSJ"), converted into IOB-style full-text chunks using the CoNLL conversion scripts for training and dev, and the WSJ-derived CoNLL-2000 full text chunking test data for testing (Tjong Kim Sang and Buchholz, 2000)

NER: the English portion of the CoNLL-2003 English Named Entity Recognition data set, for which the source data was taken from Reuters newswire articles (Tjong Kim Sang and De Meulder (2003): "Reuters")

MWE: the MWE dataset of Schneider et al. (2014b), over a portion of text from the English Web Treebank[5] ("EWT")

For all tasks other than MWE,[6] we additionally have an out-of-domain test set, in order to evaluate the out-of-domain robustness of the different word representations, with and without updating. These datasets are as follows:

POS tagging: the English Web Treebank with Penn POS tags ("EWT")

Chunking: the Brown Corpus portion of the Penn Treebank ("Brown"), converted into IOB-style full-text chunks using the CoNLL conversion scripts

NER: the MUC-7 named entity recognition corpus[7] ("MUC7")

For reproducibility, we tuned the hyperparameters with random search over the development data for each task (Bergstra and Bengio, 2012). In this, we randomly sampled 50 distinct hyperparameter sets with the same random seed for the non-updating models (i.e., the models that don't update the word representation), and sampled 100 distinct hyperparameter sets for the updating models (i.e., the models that do). For each set of hyperparameters and task, we train a model over its training set and choose the best one based on its performance on development data (Turian et al., 2010). We also tune the word representation hyperparameters — namely, the word vector size d and context window size m (distributed representations), and in the case of Brown, the number of clusters.

For the updating models, we found that the results over the test data were always inferior to those that do not update the word representations, due to the higher number of hyperparameters and small sample size (i.e., 100). Since the two-layer model of the graph transformer contains a distinct set of hyperparameters for each layer, we reuse the best-performing hyperparameter settings from the non-updating models, and only tune the hyperparameters of AdaGrad for the word representation layer. This method requires only 32 additional runs and achieves consistently better results than 100 random draws.

In order to test the impact of the volume of training data on the different models (**RQ2**), we split the training set into 10 partitions based on a base-2 log scale (i.e., the second smallest partition will be twice the size of the smallest partition), and created 10 successively larger training sets by merging these partitions from the smallest one to the largest one, and used each of these to train a model. From these, we construct learning

[4] For the MWE dataset, no such split pre-existed, so we constructed our own.

[5] https://catalog.ldc.upenn.edu/LDC2012T13

[6] Unfortunately, there is no second domain which has been hand-tagged with MWEs using the method of Schneider et al. (2014b) to use as an out-of-domain test corpus.

[7] https://catalog.ldc.upenn.edu/LDC2001T02

	Training	Development	*In-domain* Test	*Out-of-domain* Test
POS tagging	WSJ Sec. 0-18	WSJ Sec. 19–21	WSJ Sec. 22–24	EWT
Chunking	WSJ	WSJ (1K sentences)	WSJ (CoNLL-00 test)	Brown
NER	Reuters (CoNLL-03 train)	Reuters (CoNLL-03 dev)	Reuters (CoNLL-03 test)	MUC7
MWE	EWT (500 docs)	EWT (100 docs)	EWT (123 docs)	—

Table 2: Training, development and test (in- and out-of-domain) data for each sequence labelling task.

curves over each task.

For ease of comparison with previous results, we evaluate both in- and out-of-domain using chunk/entity/expression-level F1-measure ("F1") for all tasks except POS tagging, for which we use token-level accuracy ("ACC"). To test performance over OOV (unknown) tokens — i.e., the words that do not occur in the training set — we use token-level accuracy for all tasks (e.g., for Chunking, we evaluate whether the full IOB tag is correct or not), because chunks/NEs/MWEs can consist of a mixture of in-vocabulary and OOV tokens, which makes the use of chunk-based evaluation measures inappropriate.

4 Experimental Results and Discussion

We structure our evaluation by stepping through each of our five research questions (**RQ1–5**) from the start of the paper. In this, we make reference to: (1) the best-performing method both in- and out-of-domain vs. the state-of-the-art (Table 3); (2) a heat map for each task indicating the convergence rate for each word representation, with and without updating (Figure 2); (3) OOV accuracy both in-domain and out-of-domain for each task (Figure 3); and (4) visualisation of the impact of updating on word embeddings, based on t-SNE (Figure 4).

RQ1: Are the selected word embeddings better than one-hot unigram features and Brown clusters? As shown in Table 3, the best-performing method for every task except in-domain Chunking is a word embedding method, although the precise method varies greatly. Figure 2, on the other hand, tells a more subtle story: the difference between UNIGRAM and the other word representations is relatively modest, esp. as the amount of training data increases. Additionally, the difference between BROWN and the word embedding methods is modest across all tasks. So, the overall answer would appear to be: yes, word embeddings are better than unigrams when there is little training data, but they are not markedly better than Brown clusters.

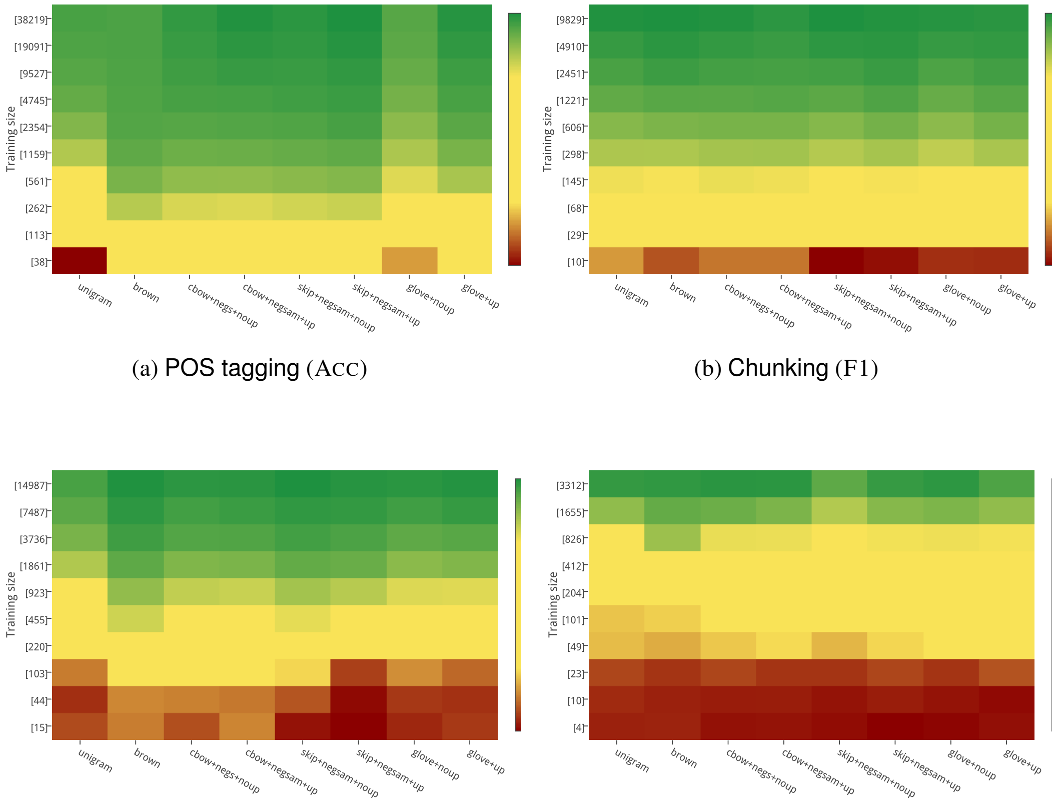
RQ2: Do word embedding features require less training data? Figure 2 shows that for POS tagging and NER, with only several hundred training instances, word embedding features achieve superior results to UNIGRAM. For example, when trained with 561 instances, the POS tagging model using SKIP-GRAM+UP embeddings is 5.3% above UNIGRAM; and when trained with 932 instances, the NER model using SKIP-GRAM is 11.7% above UNIGRAM. Similar improvements are also found for other types of word embeddings and BROWN, when the training set is small. However, all word representations perform similarly for Chunking regardless of training data size. For MWE, BROWN performs slightly better than the other methods when trained with approximately 25% of the training instances. Therefore, we conjecture that the POS tagging and NER tasks benefit more from distributional similarity than Chunking and MWE.

RQ3: Does task-specific updating improve all word embeddings across all tasks? Based on Figure 2, updating of word representations can equally correct poorly-learned word representations, and harm pre-trained representations, due to overfitting. For example, GLOVE performs substantially worse than SKIP-GRAM for both POS tagging and NER without updating, but *with* updating, the relative empirical gap between the best performing method becomes smaller. In contrast, SKIP-GRAM performs worse over the test data with updating, despite the results on the development set improving by 1%.

To further investigate the effects of updating, we sampled 60 words and plotted the changes in their word embeddings under updating, using 2-d vector fields generated using matplotlib and t-SNE (van der Maaten and Hinton, 2008). Half of the words were chosen manually to include known word clusters such as days of the week and names of countries; the other half were selected randomly. Additional plots with 100 randomly-sampled words and the top-100 most frequent words, for all the methods and all the tasks, can be found in the supplementary material and at

Task	Benchmark	*In-domain* Test set	*Out-of-domain* Test set
POS tagging (ACC)	**0.972** (Toutanova et al., 2003)	0.959 (SKIP-GRAM+UP)	0.910 (SKIP-GRAM)
Chunking (F1)	**0.942** (Sha and Pereira, 2003)	0.938 (BROWN$_{b=2000}$)	0.676 (GLOVE)
NER (F1)	**0.893** (Ando and Zhang, 2005)	0.868 (SKIP-GRAM)	0.736 (SKIP-GRAM)
MWE (F1)	0.625 (Schneider et al., 2014a)	**0.654** (CBOW+UP)	—

Table 3: State-of-the-art results vs. our best results for in-domain and out-of-domain test sets.



(a) POS tagging (ACC)

(b) Chunking (F1)

(c) NER (F1)

(d) MWE (F1)

Figure 2: Results for each type of word representation over POS tagging, Chunking, NER and MWE, optionally with updating ("+UP"). The y-axis indicates the training data sizes (on a log scale). Green = high performance, and red = low performance, based on a linear scale of the best- to worst-result for each task.

https://goo.gl/Y8bk2w. In each plot, a single arrow signifies one word, pointing from the position of the original word embedding to the updated representation.

In Figure 4, we show vector fields plots for Chunking and NER using SKIP-GRAM embeddings. For Chunking, most of the vectors were changed with similar magnitude, but in very different directions, including within the clusters of days of the week and country names. In contrast, for NER, there was more homogeneous change in word vectors belonging to the same cluster. This greater consistency is further evidence that semantic homogeneity appears to be more beneficial for NER than Chunking.

RQ4: What is the impact of word embeddings cross-domain and for OOV words? As shown in Table 3, results predictably drop when we evaluate out of domain. The difference is most pronounced for Chunking, where there is an absolute drop in F1 of around 30% for all methods, indicating that word embeddings and unigram features provide similar information for Chunking.

Another interesting observation is that updating often hurts out-of-domain performance because the distribution between domains is different. This suggests that, if the objective is to optimise performance across domains, it is best not to perform

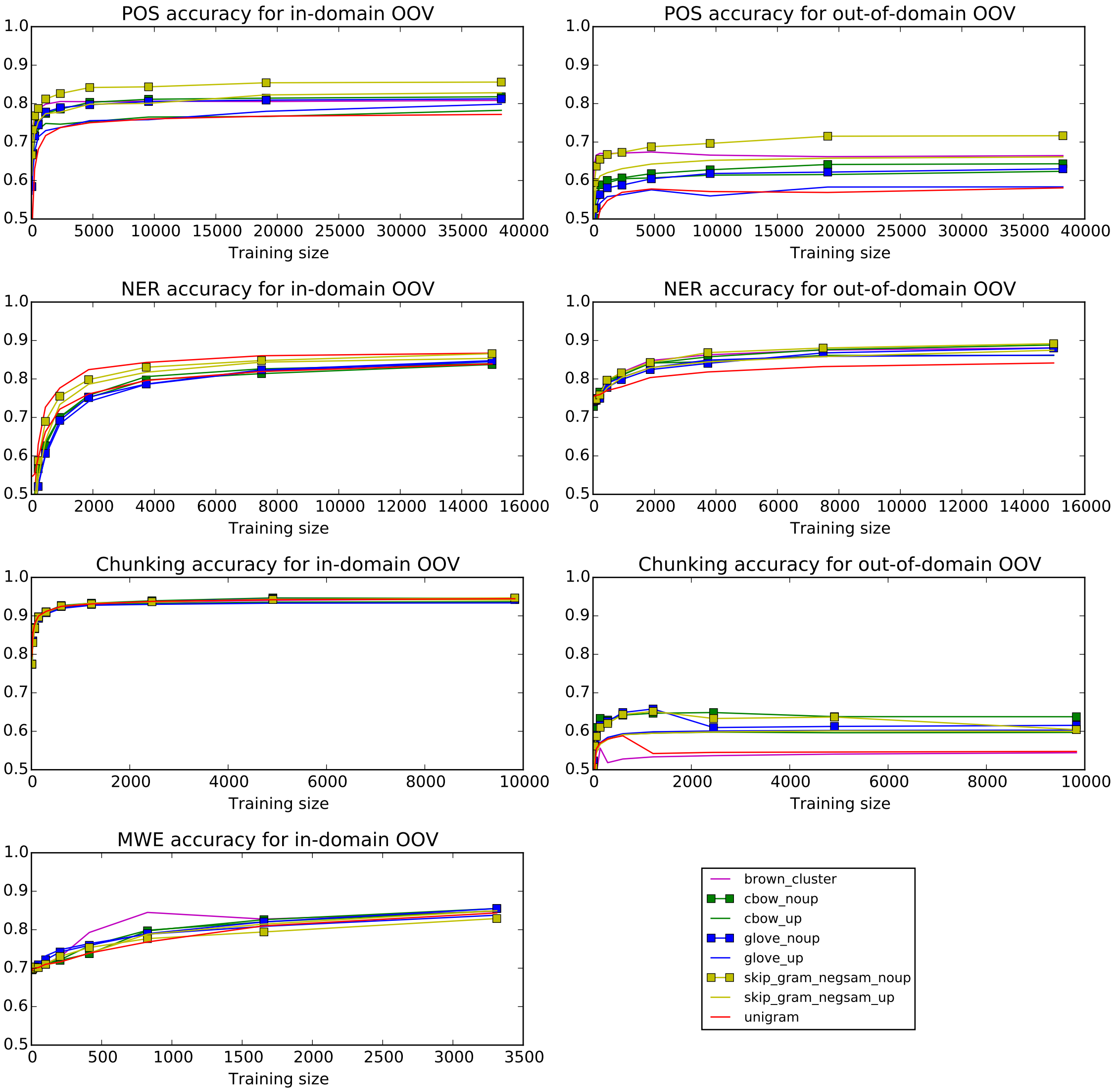
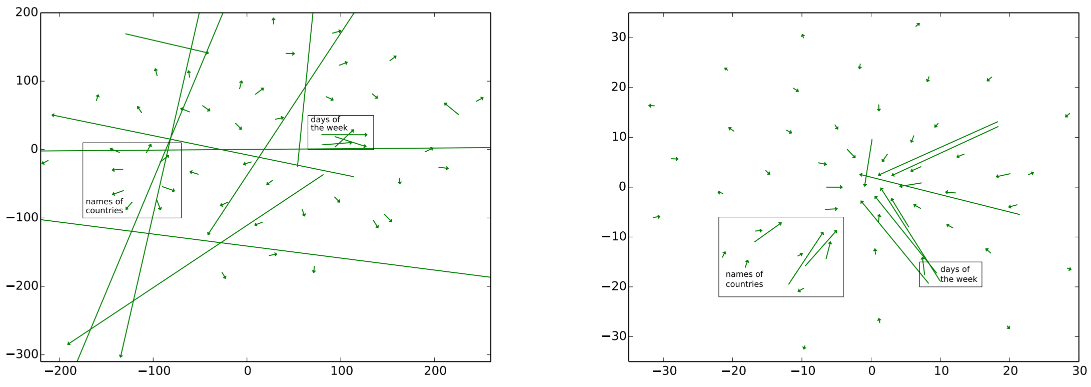


Figure 3: ACC over out-of-vocabulary (OOV) words for *in-domain* and *out-of-domain* test sets.



(a) Chunking

(b) NER

Figure 4: A t-SNE plot of the impact of updating on SKIP-GRAM

updating.

We also analyze performance on OOV words both in-domain and out-of-domain in Figure 3. As expected, word embeddings and BROWN excel in out-of-domain OOV performance. Consistent with our overall observations about cross-domain generalisation, the OOV results are better when updating is not performed.

RQ5 Overall, are some word embeddings better than others? Comparing the different word embedding techniques over our four sequence labelling tasks, for the different evaluations (overall, out-of-domain and OOV), there is no clear winner among the word embeddings — for POS tagging, SKIP-GRAM appears to have a slight advantage, but this does not generalise to other tasks.

While the aim of this paper was not to achieve the state of the art over the respective tasks, it is important to concede that our best (in-domain) results for NER, POS tagging and Chunking are slightly worse than the state of the art (Table 3). The 2.7% difference between our NER system and the best performing system is due to the fact that we use a first-order instead of a second-order CRF (Ando and Zhang, 2005), and for the other tasks, there are similarly differences in the learner and the complexity of the features used. Another difference is that we tuned the hyperparameters with random search, to enable replication using the same random seed. In contrast, the hyperparameters for the state-of-the-art methods are tuned more extensively by experts, making them more difficult to reproduce.

5 Related Work

Collobert et al. (2011) proposed a unified neural network framework that learns word embeddings and applied it to POS tagging, Chunking, NER and semantic role labelling. When they combined word embeddings with hand-crafted features (e.g., word suffixes for POS tagging; gazetteers for NER) and applied other tricks like cascading and classifier combination, they achieved state-of-the-art performance. Similarly, Turian et al. (2010) evaluated three different word representations on NER and Chunking, and concluded that unsupervised word representations improved NER and Chunking. They also found that combining different word representations can further improve performance. Guo et al. (2014) also explored different ways of using word embeddings for NER. Owoputi et al. (2013) and Schneider et al. (2014a) found that BROWN clustering enhances Twitter POS tagging and MWE, respectively. Compared to previous work, we consider *more* word representations including the most recent work and evaluate them on *more* sequence labelling tasks, wherein the models are trained with training sets of varying size.

Bansal et al. (2014) reported that direct use of word embeddings in dependency parsing did not show improvement. They achieved an improvement only when they performed hierarchical clustering of the word embeddings, and used features extracted from the cluster hierarchy. In a similar vein, Andreas and Klein (2014) explored the use of word embeddings for constituency parsing and concluded that the information contained in word embeddings might duplicate the one acquired by a syntactic parser, unless the training set is extremely small. Other syntactic parsing studies that reported improvements by using word embeddings include Koo et al. (2008), Koo et al. (2010), Haffari et al. (2011), Tratz and Hovy (2011) and Chen and Manning (2014).

Word embeddings have also been applied to other (non-sequential NLP) tasks like grammar induction (Spitkovsky et al., 2011), and semantic tasks such as semantic relatedness, synonymy detection, concept categorisation, selectional preference learning and analogy (Baroni et al., 2014; Levy and Goldberg, 2014; Levy et al., 2015).

Huang and Yates (2009) demonstrated that using distributional word representations methods (like TF-IDF and LSA) as features, improves the labelling of OOV, when test for POS tagging and Chunking. In our study, we evaluate the labelling performance of OOV words for updated vs. non-updated word embedding representations, relative to the training set and with out-of-domain data.

6 Conclusions

We have performed an extensive extrinsic evaluation of four word embedding methods under fixed experimental conditions, and evaluated their applicability to four sequence labelling tasks: POS tagging, Chunking, NER and MWE identification. We found that word embedding features reliably outperformed unigram features, especially with limited training data, but that there was relatively little difference over Brown clusters, and no one embedding method was consistently superior across the different tasks and settings. Word embeddings and Brown clusters were also found to improve out-of-domain performance and for OOV words. We expected a performance gap between the fixed and task-updated embeddings, but the observed difference was marginal. Indeed, we found that updating can result in overfitting. We also carried out preliminary analysis of the impact of updating on the vectors, a direction which we intend to pursue further.

7 Acknowledgments

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

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.

Timothy Baldwin and Su Nam Kim. 2010. Multiword expressions. In Nitin Indurkhya and Fred J. Damerau, editors, *Handbook of Natural Language Processing*. CRC Press, Boca Raton, USA, 2nd edition.

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, USA.

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

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

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.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. Technical report, Google.

Danqi Chen and Christopher D 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 2014)*, pages 740–750, Doha, Qatar.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167, Helsinki, Finland.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *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. *Journal of Machine Learning Research*, 12:2121–2159.

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 *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 281–285.

Jiang Guo, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Revisiting embedding features for simple semi-supervised learning. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 110–120, Doha, Qatar.

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

Lushan Han, Abhay L. Kashyap, Tim Finin, James Mayfield, and Johnathan Weese. 2013. UMBC EBIQUITY CORE: Semantic textual similarity systems. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics*, pages 44–52, Atlanta, USA.

Timo Honkela. 1997. Self-organizing maps of words for natural language processing applications. In *Proceedings of the International ICSC Symposium on Soft Computing*, pages 401–407, Nimes, France.

Fei Huang and Alexander Yates. 2009. Distributional representations for handling sparsity in supervised sequence-labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, pages 495–503, Suntec, Singapore.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, USA.

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 2010)*, pages 1288–1298, Cambridge, USA.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289, Williamstown, USA.

Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3(1):211–225.

Wei Li and Andrew McCallum. 2005. Semi-supervised sequence modeling with syntactic topic models. In *Proceedings of the National Conference on Artificial Intelligence*, Pittsburgh, USA.

Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1030–1038, Suntec, Singapore.

Kevin Lund and Curt Burgess. 1996. Producing high-dimensional semantic spaces from lexical cooccurrence. *Behavior Research Methods, Instruments, & Computers*, 28(2):203–208.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics*, 19(2):313–330.

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 *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

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 *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2013)*, pages 380–390, Atlanta, USA.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543, Doha, Qatar.

Fernando Pereira, Naftali Tishby, and Lillian Lee. 1993. Distributional clustering of English words. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 183–190, Columbus, USA.

Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. 2007. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th International Conference on Machine Learning*, pages 759–766, Corvallis, USA.

Radim Řehůřek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 51–55, Valetta, Malta.

Magnus Sahlgren. 2006. *The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. Ph.D. thesis, Institutionen för lingvistik.

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 *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, pages 455–461, Reykjavík, Iceland.

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*, pages 134–141, Edmonton, Canada.

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*, pages 1281–1290, Edinburgh, UK.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *Proceedings of the 4th Conference on Computational Natural Language Learning (CoNLL-2000)*, pages 127–132, Lisbon, Portugal.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the 7th Conference on Natural Language Learning (CoNLL-2003)*, pages 142–147, Edmonton, Canada.

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*, pages 173–180, Edmonton, Canada.

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*, pages 1257–1268, Edinburgh, UK.

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, Uppsala, Sweden.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.

Laurens J.P. van der Maaten and Geoffrey Hinton. 2008. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.