# Learning Word Representations from Scarce and Noisy Data with Embedding Sub-spaces

**Ramon F. Astudillo, Silvio Amir, Wang Lin, Mário Silva, Isabel Trancoso**
Instituto de Engenharia de Sistemas e Computadores Investigação e Desenvolvimento (INESC-ID)
Rua Alves Redol 9
Lisbon, Portugal
{ramon.astudillo, samir, wlin, mjs, isabel.trancoso}@inesc-id.pt

## Abstract

We investigate a technique to adapt unsupervised word embeddings to specific applications, when only small and noisy labeled datasets are available. Current methods use pre-trained embeddings to initialize model parameters, and then use the labeled data to tailor them for the intended task. However, this approach is prone to overfitting when the training is performed with scarce and noisy data. To overcome this issue, we use the supervised data to find an embedding subspace that fits the task complexity. All the word representations are adapted through a projection into this task-specific subspace, even if they do not occur on the labeled dataset. This approach was recently used in the SemEval 2015 Twitter sentiment analysis challenge, attaining state-of-the-art results. Here we show results improving those of the challenge, as well as additional experiments in a Twitter Part-Of-Speech tagging task.

## 1 Introduction

The success of supervised systems largely depends on the amount and quality of the available training data, oftentimes, even more than the particular choice of learning algorithm (Banko and Brill, 2001). Labeled data is, however, expensive to obtain, while unlabeled data is widely available. In order to exploit this fact, semi-supervised learning methods can be used. In particular, it is possible to derive word representations by exploiting word co-occurrence patterns in large samples of unlabeled text. Based on this idea, several methods have been recently proposed to efficiently estimate *word embeddings* from raw text, leveraging neural language models (Huang et al., 2012; Mikolov et al., 2013; Pennington et al., 2014; Ling et al., 2015). These models work by maximizing the probability that words within a given window size are predicted correctly. The resulting embeddings are low-dimensional dense vectors that encode syntactic and semantic properties of words. Using these word representations, Turian et al. (2010) were able to improve near state-of-the-art systems for several tasks, by simply plugging in the learned word representations as additional features. However, because these features are estimated by minimizing the prediction errors made on a generic, unsupervised, task they might be suboptimal for the intended purposes.

Ideally, word features should be adapted to the specific supervised task. One of the reasons for the success of deep learning models for language problems, is the use unsupervised word embeddings to initialize the word projection layer. Then, during training, the errors made in the predictions are backpropagated to update the embeddings, so that they better predict the supervised signal (Collobert et al., 2011; dos Santos and Gatti, 2014a). However, this strategy faces an additional challenge in noisy domains, such as social media. The lexical variation caused by the typos, use of slang and abbreviations leads to a great number of singletons and out-of-vocabulary words. For these words, the embeddings will be poorly re-estimated. Even worse, words not present on the training set will never get their embeddings updated.

In this paper, we describe a strategy to adapt unsupervised word embeddings when dealing with small and noisy labeled datasets. The intuition behind our approach is the following. For a given task, only a subset of all the latent aspects captured by the word embeddings will be useful. Therefore, instead of updating the embeddings directly with the available labeled data, we estimate a projection of these embeddings into a low dimensional sub-space. This simple method brings two fundamental advantages. On the one hand, we obtain low dimensional embeddings fitting the complexity of the target task. On the other hand, we are able to learn new representations for all the words, even if they do not occur in the labeled dataset.

To estimate the low dimensional sub-space, we propose a simple non-linear model equivalent to a neural network with one single hidden layer. The model is trained in supervised fashion on the labeled dataset, learning jointly the sub-space projection and a classifier for the target task. Using this model, we built a system to participate in the SemEval 2015 Twitter sentiment analysis benchmark (Rosenthal et al., 2015). Our submission attained state-of-the-art results without hand-coded features or linguistic resources (Astudillo et al., 2015). Here, we further investigate this approach and compare it against several state-of-the-art systems for Twitter sentiment classification. We also report on additional experiments to assess the adequacy of this strategy in other natural language problems. To this end, we apply the embedding sub-space layer to Ling et al. (2015) deep learning model for part-of-speech tagging. Even though the gains were not as significant as in the sentiment polarity prediction task, the results suggest that our method is indeed generalizable to other problems.

The rest of the paper is organized as follows: the related work is reviewed in Section 2. Section 3, briefly describes the model used to pre-train the word embeddings. In Section 4, we introduce the concept of embedding sub-space, as well as the the non-linear sub-space model for text classification. Section 5, details the experiments performed with the SemEval corpora. Section 6 describes additional experiments applying the embedding sub-space method to a Part-of-Speech tagging model for Twitter data. Finally, Section 7 draws the conclusions.

## 2 Related Work

NLP systems can benefit from a very large pool of unlabeled data. While raw documents are usually not annotated, they contain structure, which can be leveraged to learn word features. Context is one strong indicator for word similarity, as related words tend to occur in similar contexts (Firth, 1968). Approaches that are based on this concept include, Latent Semantic Analysis, where words are represented as rows in the low-rank approximation of a term co-occurrence matrix (Dumais et al., 1988), word clusters obtained with hierarchical clustering algorithms based on Hidden Markov Models (Brown et al., 1992), and continuous word vectors learned with neural language models (Bengio et al., 2003). The resulting clusters and vectors, can then be used as more generalizable features in supervised tasks, as they also provide representations for words not present in the labeled data (Bespalov et al., 2011; Owoputi et al., 2013; Chen and Manning, 2014).

A great amount of work has been done on the problem of learning better word representations from unsupervised data. However, not many studies have discussed the best ways to use them in supervised tasks. Typically, in these cases, word representations are directly used as features or to initialize the parameters of more complex models. In some tasks, this approach is however prone to overfitting. The work presented here aims to provide a simple approach to overcome this last scenario. It is thus directly related to Labutov and Lipson (2013), where a method to learn task-specific representations from general pre-trained embeddings was presented. In this work, new features were estimated with a convex objective function that combined the log-likelihood of the training data, with regularization penalizing the Frobenius norm of the distortion matrix. That is, the matrix of the differences between the original and the new embeddings. Even though the adapted embeddings performed better than the purely unsupervised features, both were significantly outperformed by a simple bag-of-words baseline.

Most other approaches, simply rely on additional training data to fine tune the embeddings for a given supervised task. In Bansal et al. (2014), better word embeddings for dependency parsing were obtained by using a corpus created to capture dependency context. This technique requires, nevertheless, of a pre-existing dependency parser or, at least a parsed corpus. For some other tasks, it is possible to collect weakly labeled corpora by making strong assumptions about the data. In Go et al. (2009) a corpus for Twitter sentiment analysis was built by assuming that tweets with positive emoticons imply positive sentiment, whereas tweets with negative emoticons imply negative sentiment. Using a similar corpus, Tang et al. (2014b) induced sentiment specific word embeddings, for the Twitter domain. The embeddings were estimated with a neural network that minimized a linear combination of two loss functions, one penalized the errors made at predicting the center word within a sequence of words, while the other penalized mistakes made at deciding the sentiment label. Weakly labeled data has also been used to refine unsupervised embeddings, by retraining them to predict the noisy labels before using the actual task-specific supervised data (Severyn and Moschitti, 2015).

## 3 Unsupervised Structured Skip-Gram Word Embeddings

Word embeddings are generally trained by optimizing an objective function that can be measured without annotations. One popular approach is to estimate the embeddings by maximizing the probability that the words within a given window size are predicted correctly. Our previous work has compared several such models, namely the skip-gram and CBOW architectures (Mikolov et al., 2013), GloVe (Pennington et al., 2014), and the structured skip-gram approach (Ling et al., 2015), suggesting that they all have comparable capabilities. Thus, in this study we only use embeddings derived with the structured skip-gram approach, a modification of the skip-gram architecture that has been shown to outperform the original model in syntax based tasks such as, part-of-speech tagging and dependency parsing.

Central to the structured skip-gram is a log linear model of word prediction. Let $w = i$ denote that a word at a given position of a sentence is the $i$-th word on a vocabulary of size $v$, and let $w^p = j$ denote that the word $p$ positions further in the sentence is the $j$-th word on the vocabulary. The structured skip-gram models the following probability:

$$p(w^p = j|w = i) \propto \exp\left(\mathbf{C}_j^p \cdot \mathbf{E} \cdot \mathbf{w}^i\right) \quad (1)$$

Here, $\mathbf{w}^i \in \{1, 0\}^{v\times1}$ is a one-hot representation of $w = i$. That is, a vector of zeros of the size of the vocabulary $v$ with a 1 on the $i$-th entry of the vector. The symbol $\cdot$ denotes internal product and $\exp()$ acts element-wise. The log-linear model is parametrized by the following matrices: $\mathbf{E} \in \mathbb{R}^{e\times v}$, is the embedding matrix, transforming the one-hot representation into a compact real valued space of size $e$, $\mathbf{C}_j^p \in \mathbb{R}^{v\times e}$ is a set of output matrices, one for each relative word position $p$, projecting the real-valued representation to a vector with the size of the vocabulary $v$. By learning a different matrix $\mathbf{C}^p$ for each relative word position, the model captures word order information, unlike the original skip-gram approach that uses only one output matrix. Finally, a distribution over all possible words is attained by exponentiating and normalizing over the $v$ possible options. In practice, negative sampling is used to avoid having to normalize over the whole vocabulary (Goldberg and Levy, 2014).

As most other neural network models, the structured skip-gram is trained with gradient-based methods. After a model has been trained, the low dimensional embedding $\mathbf{E} \cdot \mathbf{w}_i \in \mathbb{R}^{e\times1}$ encapsulates the information about each word $\mathbf{w}_i$ and its surrounding contexts. This embeding can thus be used as input to other learning algorithms to further enhance performance.

## 4 Adapting Embeddings with Sub-space Projections

As detailed in the introduction and related work, word embeddings are a useful unsupervised technique to attain initial model values or features prior to supervised training. These models can be then retrained using the available labeled data. However, even if the embeddings provide a compact real valued representation of each word in a vocabulary, the total number of parameters in the model can be rather high. If, as it is often the case, only a small amount of supervised data is available, this can lead to severe overfitting. Even if regularization is used to reduce the overfitting risk, only a reduced subset of the words will actually be present in the labeled dataset. Words not seen during training will never get their embeddings updated. Furthermore, rare words will receive very few updates, and thus their embeddings will be poorly adapted for the intended task. We propose a simple solution to avoid this problem.

### 4.1 Embedding Sub-space

Let $\mathbf{E} \in \mathbb{R}^{e\times v}$ denote the original embedding matrix obtained, e.g. with the structured skip-gram model described in Equation 1. We define the adapted embedding matrix as the factorization $\mathbf{S} \cdot \mathbf{E}$, where $\mathbf{S} \in \mathbb{R}^{s\times e}$, with $s \ll e$. We estimate the parameters of the matrix $\mathbf{S}$ using the labeled dataset, while $\mathbf{E}$ is kept fixed. In other words, we determine the optimal projection of the embedding matrix $\mathbf{E}$ into a sub-space of dimension $s$.

The idea of embedding sub-space rests on two fundamental principles:

1. With dimensionality reduction of the embeddings, the model can better fit the complexity of the task at hand or the amount of available data.

2. Using a projection, all the embeddings are indirectly updated, not only those of words present in the labeled dataset.

One question that arises from this approach, is if the estimated projection is also optimal for the words not present in the labeled dataset. We assume that the words on the labeled dataset are, to some extent, representative of the words found in the unlabeled corpus. This is a reasonable assumption since both datasets can be seen as samples drawn from the same power-law distribution. If this holds, for every *unknown* word, there will be some other word sufficiently close it in the embedding space. Consequently, the projection matrix $\mathbf{S}$ will also be approximately valid for those unseen words. It is often the case that a relatively small number of words of the labeled dataset are not present on the unlabeled corpus. These words are not represented in $\mathbf{E}$. One way to deal with this case, is to simply set the embeddings of unknown words to zero. But in this case, the embeddings will not be adapted during training. Random initializations of the embeddings seems to be helpful for tasks that have a higher penalty for missing words, although it remains unclear if better initialization strategies exist.

### 4.2 Non-Linear Embedding Sub-space Model

The concept of embedding sub-space can be applied to log-linear classifiers or any deep learning architecture that uses embeddings. We now describe an application of this method for short text classification tasks. In what follows, we will refer to this approach as Non-Linear Sub-space Embedding (NLSE) model. The NLSE can be interpreted as a simple feed-forward neural network model (Rumelhart et al., 1985) with one single hidden layer utilizing the embedding sub-space approach, as depicted in Fig. 1. Let

$$\mathbf{m} = [\mathbf{w}_1 \cdots \mathbf{w}_n] \qquad (2)$$

denote a message of $n$ words. Each column $\mathbf{w} \in \{0,1\}^{v\times 1}$ of $\mathbf{m}$ represents a word in one-hot form, as described in Section 3. Let $y$ denote a categorical random variable over $K$ classes. The NLSE model, estimates thus the probability of each possible category $y = k \in K$ given a message $\mathbf{m}$ as

$$p(y = k|\mathbf{m}) \propto \exp\left(\mathbf{Y}_k \cdot \mathbf{h} \cdot \mathbf{1}\right). \qquad (3)$$

Here, $\mathbf{h} \in \{0,1\}^{e\times n}$ are the activations of the hidden layer for each word, given by

$$\mathbf{h} = \sigma\left(\mathbf{S} \cdot \mathbf{E} \cdot \mathbf{m}\right) \qquad (4)$$

where $\sigma()$ is a sigmoid function acting on each element of the matrix. The matrix $\mathbf{Y} \in \mathbb{R}^{3\times s}$ maps the embedding sub-space to the classification space and $\mathbf{1} \in 1^{n\times 1}$ is a matrix of ones that sums the scores for all words together, prior to normalization. This is equivalent to a bag-of-words assumption. Finally, the model computes a probability distribution over the $K$ classes, using the *softmax* function.

Compared to a conventional feed-forward network employing embeddings for natural language classification tasks, two main differences arise. First, the input layer is factorized into two components, the embeddings attained in unsupervised form, $\mathbf{E}$, and the projection matrix $\mathbf{S}$. Second, the size of the sub-space, in which the embeddings are projected, is much smaller than that of the original embeddings with typical reductions above one order of magnitude. As usual in this kind of models, all the parameters can be trained with gradient methods, using the backpropagation update rule.

## 5 NLSE for Twitter Sentiment Analysis

In this section, we apply the NLSE model to the message polarity classification task proposed by SemEval, for their well-known Twitter sentiment analysis challenge (Nakov et al., 2013). Given a message, the goal is to decide whether it expresses a positive, negative, or neutral sentiment. Most of the top performing systems that participated in this challenge, relied on linear classification models and the bag-of-words assumption, representing messages as sparse vectors of the size of the vocabulary. In the case of social media, this approach is particularly inefficient, due to the large vocabularies necessary to account for all the lexical variation found in this domain. Thus, these models

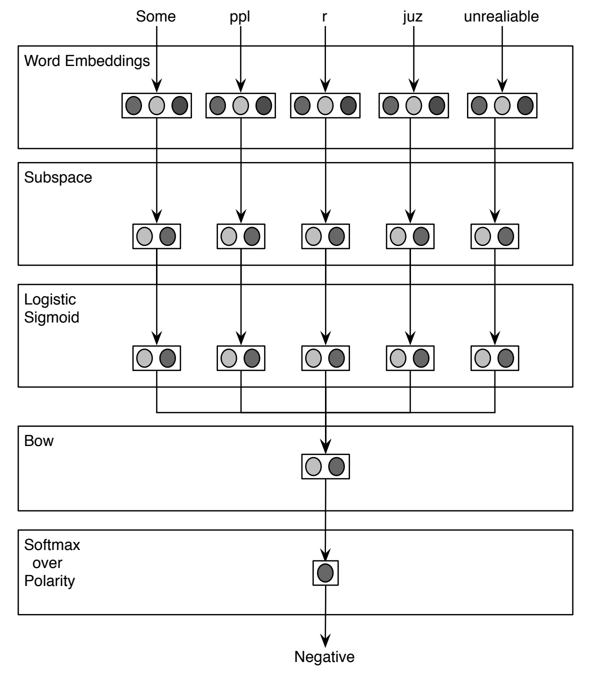


Figure 1: Illustration of the NLSE model, applied to sentiment polarity prediction.

| | Positive | Neutral | Negative |
|---|---|---|---|
| Development | 3230 | 4109 | 1265 |
| Tweets 2015 | 1032 | 983 | 364 |
| Tweets 2014 | 982 | 669 | 202 |
| Tweets 2013 | 1572 | 1640 | 601 |

Table 1: Number of examples per class in each SemEval dataset. The first row shows the training data; the other rows are sets used for testing.

need to be enriched with additional hand-crafted features that try to capture more discriminative aspects of the content, most of which require external tools (e.g., part-of-speech taggers and parsers) or linguistic resources (e.g., dictionaries and sentiment lexicons) (Miura et al., 2014; Kiritchenko et al., 2014). With the embedding sub-space approach, however, we are able to attain state-of-the-art performance while requiring only minimal processing of the data and few hyperparameters. To make our results comparable to other systems for this task, we adopted the guidelines from the benchmark. Our system was trained and tuned using only the development data. The evaluation was performed on the test sets, shown in Table 1, and we report the results in terms of the average F-measure for the positive and negative classes.

## 5.1 Experimental Setup

The first step of our approach requires a corpus of raw text for the unsupervised pre-training of the embedding matrix **E**. We resorted to the corpus of 52 million tweets used in (Owoputi et al., 2013) and the tokenizer described in the same work. The messages were previously pre-processed as follows: lower-casing, replacing Twitter user mentions and URLs with special tokens and reducing any character repetition to at most 3 characters. Words occurring less than 40 times in the corpus were discarded, resulting in a vocabulary of around 210,000 types. Then, a modified version of the `word2vec` tool[1] was used to compute the word embeddings, as described in Section 3. The window size and negative sampling rate were set to 5 and 25 words, respectively, and embeddings of 50, 200, 400 and 600 dimensions were built.

Our system accepts as input a sentence represented as a matrix, obtained by concatenating the *one-hot* vectors that represent each individual word. Therefore, we first performed the aforementioned normalization and tokenization steps and then, converted each tweet into this representation. The development set was split into $80\%$ for parameter learning and $20\%$ for model evaluation and selection, maintaining the original relative class proportions in each set. All the weights were initialized uniformly at random, as proposed in (Glorot and Bengio, 2010). The model was trained with conventional Stochastic Gradient Descent (Rumelhart et al., 1985) with a fixed learning rate of $0.01$, and the weights were updated after each message was processed. Variations of learning rate to smaller values, e.g. $0.005$, were considered but did not lead to a clear pattern. We explored different configurations of the hyperparameters $e$ (embedding size) and $s$ (sub-space size). Model selection was done by early stopping, i.e., we kept the configuration with best F-measure on the evaluation set after 5-8 iterations.

## 5.2 Results

In general, the NLSE model showed consistent and fast convergence towards the optimum in very few iterations. Despite using class log-likelihood as training criterion, it showed good performance in terms of the average F-measure for positive and negative sentiments. We found that all embedding sizes yield comparable performances, al-

[1]`https://github.com/wlin12/wang2vec`

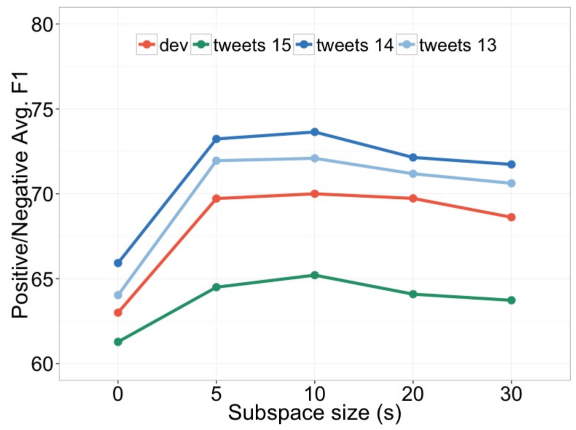


Figure 2: Average F-measure on the SemEval test sets varying with embedding sub-space size s. Sub-space size 0 used to denote the baseline (log-linear model).

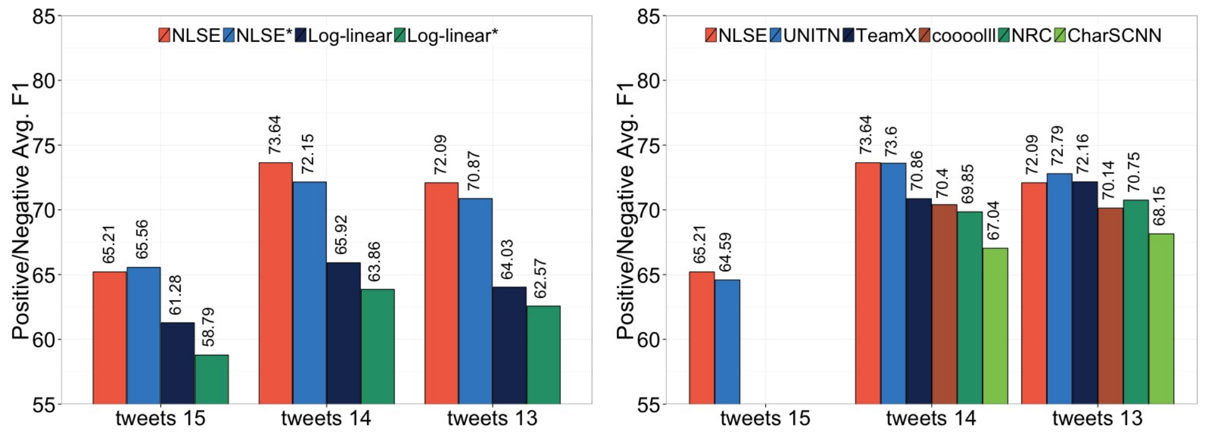
though larger embeddings tend to perform better. Therefore, we only report results obtained with the 600 dimensional vectors. In Figure 2, we show the variation of system performance with sub-space size $s$. The baseline is a log-linear using the embeddings in $\mathbf{E}$ as features. As it can be seen, the performance is sharply improved when the embedding sub-spaces are used. By choosing different values of $s$, we can adjust the model to the complexity of the task and the amount of labeled data available. Given the small size of the training set, the best results were attained with the use of smaller sub-spaces, in the range of 5-10 dimensions.

Figure 3, presents the main results of the experimental evaluation. As baselines, we considered two simple approaches: LOG-LINEAR, which uses the unsupervised embeddings directly as features in a log-linear classifier, and LOG-LINEAR*, also using the unsupervised embeddings as features in a log-linear classifier, but updating the embeddings with the training data. These baselines, were compared against two variations of the non-linear sub-space embedding model: NLSE, where we only train the $\mathbf{S}$ and $\mathbf{Y}$ weights while the embeddings are kept fixed, and NLSE*, where we also update the embedding matrix during training. For these experiments, we set $s = 10$. The results in Figure 3a, show that our model largely outperforms the simpler baselines. Furthermore, we observe that updating the embeddings always leads to inferior results. This suggests that pre-computed embeddings should be kept fixed, when little labeled data is available to re-train them.

## Comparison with the state-of-the-art

We now compare the NLSE model with state-of-the-art systems, including the best submissions to previous SemEval benchmarks. We also include two other approaches that are related to the one here proposed, where a neural network initialized with pre-trained word embeddings is used to learn relevant features. Specifically, we compare the following systems:

- NRC (Kiritchenko et al., 2014), a support vector machine classifier with a large set of hand-crafted features, including word and character n-grams, brown clusters, POS tags, morphological features, and a set of features based on five sentiment lexicons. Most of the performance was due to the combination of these lexicons. This was the top system in the 2013 edition of SemEval.
- TEAMX (Miura et al., 2014), a logistic regression classifier using a similar set of features. Additional features based on two different POS taggers and a word sense disambiguator were also included in the model. This approach attained the highest ranking in the 2014 edition.
- CHARSCNN (dos Santos and Gatti, 2014b), a deep learning architecture with two convolutional layers that exploit character-level and word-level information. The features are extracted by converting a sentence into a sequence of word embeddings, and the individual words into sequences of character embeddings. Convolution filters followed by $max$ pooling are applied to these sequences, to produce fixed size vectors. These vectors are then combined and transfered to a set of non-linear activation functions, to generate more complex representations of the input. The predictions, based on these learned features are computed with a $softmax$ classifier.
- COOOOLLL (Tang et al., 2014a), a support vector machine classifier that leverages the sentiment specific word embeddings, discussed in Section 2. The embeddings are also processed with a convolution filter, but the output of this operation is used to produce three representations obtained with different strategies, namely with $max$, $min$ and



(a) Comparison of two baselines with two variations of the NLSE model

(b) Performance of state-of-the-art systems for Twitter sentiment prediction

Figure 3: Average F-measure on the SemEval test sets

$average$ pooling. The final feature vector is obtained by concatenating these representations and Kiritchenko et al. (2014) feature set.

- UNITN (Severyn and Moschitti, 2015), another deep convolutional neural network that jointly learns internal representations and a $softmax$ classifier. The network is trained in three steps: (i) unsupervised pre-training of embeddings, (ii) refinement of the embeddings using a weakly labeled corpus, and (iii) fine tuning the model with the labeled data from SemEval. It should be noted that the system was trained with a labeled corpus 65% larger than ours[2]. This system made the best submission on the 2015 edition of the benchmark.

The results in Figure 3b, show that despite being simpler and requiring less resources and labeled data, the NLSE model is extremely competitive, even outperforming most other systems, in predicting the sentiment polarity of Twitter messages.

## 6 Generalization to Other Tasks

While the embedding sub-space method works well for the sentiment prediction task, we would like to know its impact in other settings that are known to benefit from unsupervised embeddings. Thus, we decided to replicate the part-of-speech tagging work in (Ling et al., 2015), where pre-training embeddings have been shown to improve the quality of the results significantly.

### 6.1 Sub-space Window Model

Part-of-speech tagging is a word labeling task, where each word is to be labeled with its syntactic function in the sentence. More formally, given an input sentence $w_1, \ldots, w_n$ of $n$ words, we wish to predict a sequence of labels $y_1, \ldots, y_n$, which are the POS tags of each of the words. This task is scored by the ratio between the number of correct labels and the number of words to be labeled.

We modified (Collobert et al., 2011) window model, to include the sub-space matrix $\mathbf{S}$. The probability of labeling the word $w_t$ with the POS tag $k$ is given by

$$p(y = k|\mathbf{m}_{t-p}^{t+p}) \propto \exp\left(\mathbf{Y}_k \cdot \mathbf{h}_t + \mathbf{b}\right), \quad (5)$$

where

$$\mathbf{m}_{t-p}^{t+p} = [\mathbf{w}_{t-p} \cdots \mathbf{w}_t \cdots \mathbf{w}_{t+p}] \quad (6)$$

denotes a context window of words around the $t$-th word, with a total span of $2p + 1$ words. $\mathbf{h}_t$ denotes the activations of a hidden layer given by

$$\mathbf{h}_t = \tanh\left(\mathbf{H} \cdot \begin{bmatrix} \mathbf{S} \cdot \mathbf{E} \cdot \mathbf{w}_{t+p} \\ \cdots \\ \mathbf{S} \cdot \mathbf{E} \cdot \mathbf{w}_t \\ \cdots \\ \mathbf{S} \cdot \mathbf{E} \cdot \mathbf{w}_{t-p} \end{bmatrix}\right). \quad (7)$$

Here $\tanh$ denotes the hyperbolic tangent, acting element-wise. Aside from embedding $\mathbf{E}$ and

[2]The UNITN system was trained with around 11,400 labeled examples, whereas we used only 6,900.

sub-space $\mathbf{S}$ matrices, the model is parametrized by the weights $\mathbf{H} \in \mathbb{R}^{h \times ps}$ and $\mathbf{Y} \in \mathbb{R}^{v \times h}$ as well as a bias $\mathbf{b} \in \mathbb{R}^{v \times 1}$.

Note that if $\mathbf{S}$ is set to the identity matrix, this would be equivalent to the original Collobert et al. (2011) model.

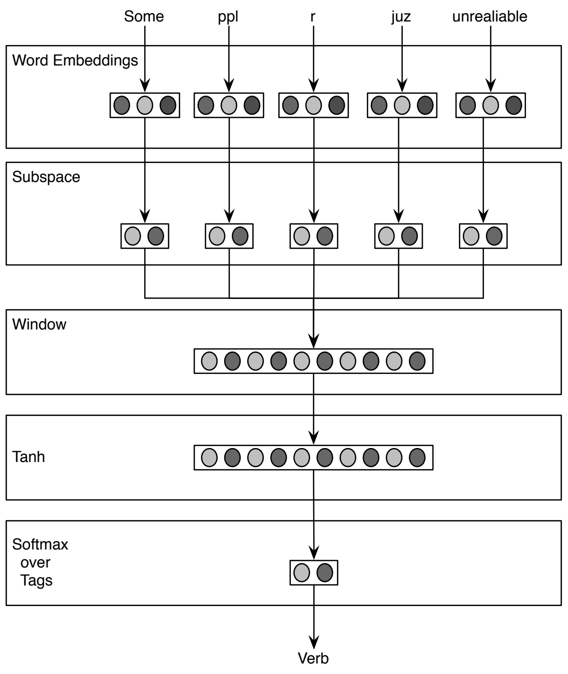


Figure 4: Illustration of the window model by (Collobert et al., 2011) using a sub-space layer.

## 6.2 Experiments

Tests were performed in Gimpel et al. (2011) Twitter POS dataset, which uses the universal POS tag set composed by 25 different labels (Petrov et al., 2012). The dataset contains 1000 annotated tweets for training, 327 tweets for tuning and 500 tweets for testing. The number of word tokens in these sets are 15000, 5000 and 7000, respectively. There are 5000, 2000 and 3000 word types.

Once again, we initialized the embeddings with unsupervised pre-training using the structured skip-gram approach. As for the hyperparameters of the model, we used embeddings with $e = 50$ dimensions, a hidden layer with $h = 200$ dimensions and a context of $p = 2$ as used in (Ling et al., 2015). Training employed mini-batch gradient descent, with mini batches of $100$ sentences and a momentum of $0.95$. The learning rate was set to $0.2$. Finally, we used early stopping by choosing the epoch with the highest accuracy in the tuning set. As for the sub-space layer size, we tried three different hyperparameterizations: 10, 30 and 50 dimensions.

## 6.3 Results

Figure 5 displays the results. Using the setup that led to the best results in the sentiment prediction task (FIX), that is, fixing $\mathbf{E}$ and updating $\mathbf{S}$, leads to lower accuracies than the baseline (TRAIN-ALL, $s = 0$). We also see that different values of $s$ do not have a very strong impact in the final results.

Sentiment polarity prediction and POS tagging differ in multiple aspects and there may be more than one reason for this poorer performance. One particularly relevant aspect, in our opinion, is the way words that have no pre-trained embedding are treated. In the case of sentiment prediction, these words were set to having and embedding of zero. This fits the use of the bag-of-words assumption and the fact that only one label is produced per message, as there are many other words to draw evidence from. In the case of POS tagging a hypothesis must be drawn for each word, using a shorter context. Thus, ignoring a word means that context is used instead, which is a frequent cause of errors.

One way around this problem would be to update the parameters of $\mathbf{S}$ and $\mathbf{E}$, but this leads to results similar to the experiment without the sub-space projections (TRAIN-ALL). This is expected as the sub-space layer was designed to work on fixed word embeddings, if these are updated its benefits are lost. Thus, we solve this problem by fixing all the embeddings, except for the word types not found in the pre-training corpus. That is, instead of leaving the unknown words as the zero vector, we use the labeled data to learn a better representation. Using this setup (TRAIN-OOV), we can obtain a small but consistent improvement over the baseline. While these improvements are not significant, as this task is not as prone to overfitting as in sentiment analysis, this is a good check of the validity of our method.

## 7 Conclusions

We presented a new approach to use unsupervised word embeddings based on the idea of finding a sub-space projection of the embeddings for a given task. This approach offers two main advantages. On the one hand, it allows to indirectly update embeddings unseen during training. On the other

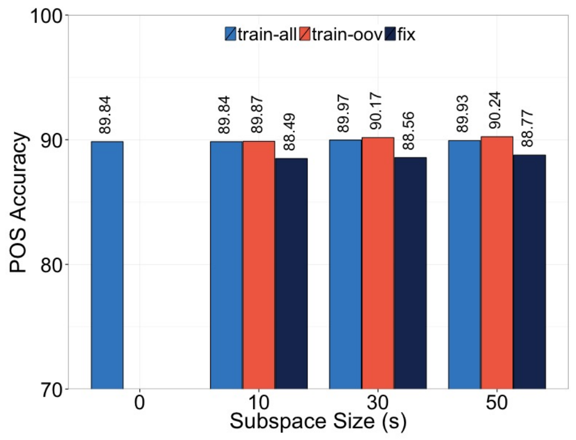


Figure 5: Results for the part-of-speech task on the ARK POS dataset, for different strategies to update the embeddings and with variations of the sub-space size. Sub-space size 0 used to denote the baseline (window model).

hand, it reduces the number of model parameters to fit the complexity of the task. These properties make this method particularly useful for the cases where only small amounts of noisy data are available to train the model.

Experiments on the SemEval challenge corpora validated these ideas, showing that such a simple approach can attain state-of-the-art results comparable with the best systems of past SemEval editions and often outperforming them in all datasets. It should be noted that this is attained while keeping the original embedding matrix $\mathbf{E}$ fixed and only learning the projection $\mathbf{S}$ with the supervised data. Additional experiments on the Twitter POS tagging task indicate however that, the technique is not always as effective as in the sentiment classification task. One possible explanation for the different behavior is the use of embeddings of zeros for words without pre-trained embedding. It is plausible that this has a stronger effect on the POS tagging task. Another aspect to be taken into account is the fact that both tasks could have a different complexity which would explain why adapting $\mathbf{E}$ in the POS taks yields better results. Optimality of the embeddings for each of the tasks might also come into play here.

The implementation of the proposed method and our Twitter Sentiment Analysis system has been made publicly available[3].

[3]`https://github.com/ramon-astudillo/NLSE`

## Acknowledgments

This work was partially supported by Fundação para a Ciência e Tecnologia (FCT), through contracts UID/CEC/50021/2013, EXCL/EEI-ESS/0257/2012 (DataStorm), research project with reference UTAP-EXPL/EEI-ESS/0031/2014, EU project SPEDIAL (FP7 611396), grant number SFRH/BPD/68428/2010 and Ph.D. scholarship SFRH/BD/89020/2012.

## References

Ramon F. Astudillo, Silvio Amir, Wang Ling, Bruno Martins, Mário Silva, and Isabel Trancoso. 2015. Inesc-id: Sentiment analysis without hand-coded features or liguistic resources using embedding subspaces. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, SemEval '2015, Denver, Colorado, June. Association for Computational Linguistics.

Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 26–33. Association for Computational Linguistics.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155.

Dmitriy Bespalov, Bing Bai, Yanjun Qi, and Ali Shokoufandeh. 2011. Sentiment classification based on supervised latent n-gram analysis. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 375–382. ACM.

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. *Comput. Linguist.*, 18(4):467–479, December.

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)*, pages 740–750.

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.

Cicero dos Santos and Maira Gatti. 2014a. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 69–78, Dublin, Ireland, August. Dublin City University and Association for Computational Linguistics.

Cıcero Nogueira dos Santos and Maıra Gatti. 2014b. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING), Dublin, Ireland*.

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. ACM.

John Rupert Firth. 1968. *Selected papers of JR Firth, 1952-59*. Indiana University Press.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, pages 249–256.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, pages 1–12.

Yoav Goldberg and Omer Levy. 2014. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 873–882, Stroudsburg, PA, USA. Association for Computational Linguistics.

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M Mohammad. 2014. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, pages 723–762.

Igor Labutov and Hod Lipson. 2013. Re-embedding words. In *Proceedings of the 51st annual meeting of the ACL*, pages 489–493.

Wang Ling, Chris Dyer, Alan Black, and Isabel Trancoso. 2015. Two/too simple adaptations of word2vec for syntax problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *27th Annual Conference on Neural Information Processing Systems*.

Yasuhide Miura, Shigeyuki Sakaki, Keigo Hattori, and Tomoko Ohkuma. 2014. Teamx: A sentiment analyzer with enhanced lexicon mapping and weighting scheme for unbalanced data. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 628–632, Dublin, Ireland, August. Association for Computational Linguistics.

Preslav Nakov, Zornitsa Kozareva, Alan Ritter, Sara Rosenthal, Veselin Stoyanov, and Theresa Wilson. 2013. Semeval-2013 task 2: Sentiment analysis in twitter.

Olutobi Owoputi, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *In Proceedings of NAACL*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 12.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. European Language Resources Association (ELRA), may.

Sara Rosenthal, Preslav Nakov, Svetlana Kiritchenko, Saif M Mohammad, Alan Ritter, and Veselin Stoyanov. 2015. Semeval-2015 task 10: Sentiment analysis in twitter. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, SemEval '2015, Denver, Colorado, June. Association for Computational Linguistics.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1985. Learning internal representations by error propagation. Technical report, DTIC Document.

Aliaksei Severyn and Alessandro Moschitti. 2015. Unitn: Training deep convolutional neural network for twitter sentiment classification. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, SemEval '2015, Denver, Colorado, June. Association for Computational Linguistics.

Duyu Tang, Furu Wei, Bing Qin, Ting Liu, and Ming Zhou. 2014a. Coooolll: A deep learning system for twitter sentiment classification. *SemEval 2014*, page 208.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014b. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1555–1565.

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.