Learning Distributed Representations for Multilingual Text Sequences

Hieu Pham **Minh-Thang Luong** **Christopher D. Manning**
Computer Science Department Stanford University, Stanford, CA, 94305
{hyhieu,lmthang,manning}@stanford.edu

Abstract

We propose a novel approach to learning distributed representations of variable-length text sequences in multiple languages simultaneously. Unlike previous work which often derive representations of multi-word sequences as weighted sums of individual word vectors, our model learns distributed representations for phrases and sentences as a whole. Our work is similar in spirit to the recent paragraph vector approach but extends to the bilingual context so as to efficiently encode meaning-equivalent text sequences of multiple languages in the same semantic space. Our learned embeddings achieve state-of-the-art performance in the often used crosslingual document classification task (CLDC) with an accuracy of 92.7 for English to German and 91.5 for German to English. By learning text sequence representations as a whole, our model performs equally well in both classification directions in the CLDC task in which past work did not achieve.

1 Introduction

Distributed representations of words, also known as word embeddings, are critical components of many neural network based NLP systems. Such representations overcome the sparsity of natural languages by representing words with high-dimensional vectors in a continuous space. These vectors encode semantic information of words, leading to success in a wide range of tasks, such as sequence tagging, sentiment analysis, and parsing (Collobert et al., 2011; Maas et al., 2011; Socher et al., 2013a; Chen and Manning, 2014). As a natural extension, being able to learn representations for larger language structures such as phrases or sentences, has also been of interest to the community lately, for instance (Socher et al., 2013b; Le and Mikolov, 2014).

In the multilingual context, most of the recent work in bilingual representation learning such as (Klementiev et al., 2012; Mikolov et al., 2013b; Zou et al., 2013; Hermann and Blunsom, 2014; Kočiský et al., 2014; Gouws et al., 2014) only focus on learning embeddings for words and use simple functions, e.g., idf-weighted sum, to synthesize representations for larger text sequences from their word members. In contrast, our work aims to learn representations for phrases and sentences as a whole so as to represent non-compositional meanings.

In essence, we extend the paragraph vector approach proposed by Le and Mikolov (2014) to the bilingual context to efficiently encode meaning-equivalent multi-word sequences in the same semantic space. Our method only utilizes parallel data and eschews the use of word alignments. When tested on the often used crosslingual document classification (CLDC) tasks, our learned embeddings yield state-of-the-art performance with an accuracy of 92.7 for English to German and 91.5 for German to English. One notable feature of our model is that it performs equally well in both classification directions in the CLDC task in which past work did not achieve as we detail later in the experiment section.

2 Related work

Word representations – Work in learning distributed representations for words can largely be

Proceedings of NAACL-HLT 2015, pages 88–94,
Denver, Colorado, May 31 – June 5, 2015.

grouped into two categories: (a) pseudo-supervised methods which make use of properties in the unannotated training data as supervised signals and (b) task-specific approaches that utilizes annotated data to learn a prediction task. For the former, word embeddings are often part of neural language models that learn to predict next words given contexts by either minimizing the cross-entropy (Bengio et al., 2003; Morin, 2005; Mnih and Hinton, 2009; Mikolov et al., 2010; Mikolov et al., 2011) or maximizing the ranking margins (Collobert and Weston, 2008; Huang et al., 2012; Luong et al., 2013). Representatives for the latter include (Collobert and Weston, 2008; Maas et al., 2011; Socher et al., 2013a) which finetune embeddings for various tasks such as sequence labelling, sentiment analysis, and constituent parsing.

Larger structure representations – Learning distributed representation for phrases and sentences is harder because one needs to learn both the compositional and non-compositional meanings beyond words. A method that learns distributed representations of sentences, which is closely related to our approach, is the paragraph vector by Le and Mikolov (2014). The method attempts to predict words in N-grams of a sentence, given the same shared sentence vector. Errors are backpropagated to train not only the word vectors but also the sentence vector. This method has an advantage that it only requires training data to be sequences of words unlike other work that require annotated data such as parse trees (Socher et al., 2013b; Socher et al., 2013a).

Multilingual embedding – Previous work to learning multilingual distributed representations, often optimize for a joint objective consisting of several monolingual components, such as neural language models, and a bilingual component to tie representations across languages together. The bilingual objective varies through different approaches and can be formulated as either a multi-task learning objective (Klementiev et al., 2012), a translation probability (Kočiský et al., 2014), or an L_2 distance of various forms between corresponding words (Mikolov et al., 2013b; Zou et al., 2013; Gouws et al., 2014).

The work of Hermann and Blunsom (2014) and Chandar A P et al. (2014) are similar to our work in eliminating the monolingual components and just training a model with bilingual objective to pull distributed representations of parallel sentences together. These approaches, however, only use simple bag-of-word models to compute sentence representations and has a potential disadvantage that it is hard to capture the non-compositional meanings of sentences. Instead, we learn representations for text sequences as a whole, similar to Le and Mikolov (2014), but in the bilingual context.

3 Joint-space bilingual embedding

In this section, we describe our method to learn the distributed representations of sentences from two languages given a parallel corpus. Our learned representations have the property that sequences of words with equivalent meanings across different languages will have their representations clustered together in the shared semantic space. We call this property the clustering constraint.

Our method is based on the following assumptions observed by (Le and Mikolov, 2014): the distributed representation vector of a sequence of words can contribute its knowledge to predict the N-grams in the sequence, and conversely, if a vector can contribute well to the task, then one can think of it as the representation of the sequence. Since the assumption is not made specific to any language, we generalize it to learn the distributed representation of word sequences in multiple languages. However, instead of duplicating the representations to have one vector per sentence per language, we simply force parallel sentences in the languages of consideration to share only one vector. This allows us to avoid a bilingual term in our learning objective function to cluster the corresponding vectors together.

Figure 1 illustrates the architecture of our model. Each word in each language is associated with a D-dimensional vector, whereas each parallel sentence is tied to the same sentence vector of dimension P. These word and sentence vectors are used to predict N-grams in both of the sentences. More precisely, suppose that $s_1, s_2, ..., s_S$ and $t_1, t_2, ..., t_T$ are our two parallel sentences that share the same sentence representation v. For every N-gram $[w_{i-N+1}, w_{i-N+2}, ..., w_i]$, where w can be either s or t, our model computes the N-gram probability as follows

$$p(w_i|w_{i-N+1}, ..., w_{i-1}) = p(w_i|f) \qquad (1)$$

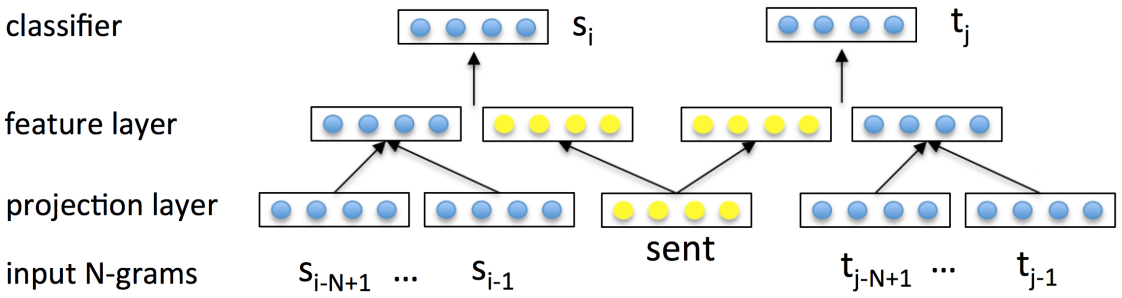

Figure 1: Our architecture to learn bilingual distributed representations of sentences. *sent* is the shared context that contributes to predicting N-grams in both sentences.

where f is a feature vector computed based on the N-gram and the shared sentence vector v.

There are several ways to compute f. As proposed in (Le and Mikolov, 2014), one can either take the average of v and the word embeddings of $w_{i-N+1}, w_{i-N+2}, ..., w_{i-1}$ or concatenate them to form f. In the former "average" approach, one always needs $D = P$, which implies that the contribution of v is less impactful as it needs to compete with the other $(N-1)$ word representations in the average term. In the latter "concatenate" approach, the dimension of f is $P + (N-1) \times D$, which suggests that the model cannot afford to have large word embedding size or longer N-grams. To overcome both problems, we propose to concatenate v with the sum of the word vectors in each of the N-grams. More precisely

$$f = \left[v; \sum_{j=i-N+1}^{i-1} w_j \right] \tag{2}$$

This hybrid approach allows us the freedom to tune D and P for our purpose.

There are also numerous approaches for the classifier that predicts the next word. However, to optimize for efficiency, we narrow our choices to the factorized multiclass classifier, also known as the hierarchical softmax (Morin, 2005). The words in the vocabulary of each language are represented as leaf-nodes of a binary tree. Each node n of the tree has a vector v_p whose dimension is equal to that of the feature vector f. These vectors encode the model's belief whether a f belongs to the left or the right child of n

$$p(\text{go left}|\text{node } n, f) = \sigma\left(f^T \cdot v_n\right) \tag{3}$$

where $\sigma(\cdot)$ is the logistic sigmoid function and v_n is a vector associated to the node n. The probability of seeing a word is then factored into the product of probabilities of the node along the path from the tree's root to the node corresponding to it.

At training time, pairs of parallel sentences are shown to the model for several epochs. The model maintains the shared sentence vectors and updates them, along with the word vectors and hierarchical softmax parameters of both languages, to minimize the cross-entropy prediction error

$$J = -\sum_{(s,t)} \log p(s,t), \tag{4}$$

where the probability of the pair of sentences (s,t) is computed simply based on the Markov assumption

$$p(s,t) = \prod_{i=1}^{S} p(s_i|s_{i-N+1}, ..., s_{i-1}) \times \prod_{j=1}^{T} p(t_j|t_{j-N+1}, ..., t_{j-1}) \tag{5}$$

At test time, the model is given one sentence in one of the languages it has been trained on. To compute the representation of that sentence, we randomly initialize a vector and train it in the same setting as above, but to predict only N-grams from one sentence. We update only the sentence vector; other parameters are preserved. We want to emphasize that due to the random initialization, the sentence embeddings computed by our model are not deterministic, in the sense that if the model sees a sentence twice, it is possible that two different embeddings will be learned for the same sentence. This

might potentially be a cause of nondeterminism if other models attempt to learn or classify based on these sentence vectors. However, our training objective for each N-gram has the form

$$J_{N\text{-gram}} = -\sum_{n} \log\left(\sigma\left(f^T \cdot v_n\right)\right), \qquad (6)$$

Since $\sigma(\cdot)$ is log-concave, our training objective is convex. This guarantees a global minimum sentence embedding vector to which our sentence vectors would converge. Moreover, at train time, the model has been trained to minimize the prediction errors of pairs of sentences that share the same sentence vector, its parameters have adapted to this manner. Hence, at test time, although two sentence vectors are learned independently, one can expect that they converge to close points in the shared semantic space.

4 Experiments

4.1 Training data and procedures

We attempt to learn the distributed representation for arbitrary sequences of words in English and German. We train our model using the `Europarl v7` multilingual corpora (Koehn, 2005), in particular the English-German corpus. The corpus consists of multilingual parliament documents automatically aligned into $1.8M$ equivalent pairs of sentences. We preprocess the corpus by filtering out the tokens that appear less than 5 times and desegment the German compound words. This leads to the final set of $43K$ English words and $95K$ German words.

Parameters of our model are updated using a gradient-based method. While for each pair of sentences, the prediction and N-grams and parameter updates are performed in sequence, our training implementation uses the multithreading approach to train through pairs of sentences in our training corpus in parallel and updates parameters with asynchronous gradient descent. Since our model predicts N-gram probabilities, we tune our hyperparameters, including P, D and the learning rate, based on the model's perplexity on the `newstests` development data provided by the Workshop in Machine Translation[1].

[1] `http://www.statmt.org/wmt15/`

At the beginning of our training procedure, we use `word2vec` (Mikolov et al., 2013a) to guide our hierarchical softmax trees. In particular, first we precompute the distributed representations for all the tokens in all the languages and run the K-Means algorithm to classify our word vectors into C classes based on L_2 distance. Then, we sort each language's vocabulary into contiguous strides of the same class. Finally, we construct our hierarchical softmax tree as the weight-balanced binary tree on each of the sorted vocab. The resulted hierarchical softmax trees thus have the semantic information about the cluster of words held by each of its nodes, similar to the WordNet taxanomy tree (Fellbaum, 1998).

We performed experiments with different settings for the model's architecture, such as the dimension P of sentence vectors, D of word vectors, and N-gram length N, and the learning rate. Our finding is that $P \approx 5D$ generally gives the best performance. Also we used the start learning rate of 0.0001 which decreases as the model is trained for more epochs. We trained all models for 50 epochs. In Section 5, we will discuss the effect of the parameters P and D. Following, we report our best experiment results, with $P = 500$, $D = 100$, $N = 7$ and $C = 500$.

4.2 Document classification on RCV1/RCV2

We test the learned bilingual distributed representations on the English-German Cross-Lingual Document Classification (CLDC, henceforth), proposed by (Klementiev et al., 2012). The corpus consists of documents from Reuter, written in English and German, annotated into 4 categories: Corporate/Industrial, Economics, Government/Social, and Markets. The documents are separated into $1K$ training documents and $5K$ test documents for each language. Each document in the dataset consist of only a few sentences, so the data is similar to the training data that our model has been trained on. The learned models are required to provide the distributed representations of all these documents, which are then passed to a perceptron algorithm to learn from training data and classify test data. The key challenge is that the perceptron algorithm has to learn in English and classify in Germen (`en`→`de`) or vice versa (`de`→`en`). To make the learning problem feasible, the document compositional model must

satisfy the clustering constraint mentioned in *Section 3*.

As in (Klementiev et al., 2012), the CLDC dataset was proposed to evaluate embeddings of words only, so we follow the author in using the document compositional model where the document vector is computed by taking the sum of all embeddings of the words that appear in the document, weighted by their inverse document frequencies (idf). We refer to this method as `para_sum`. It demonstrates that the English and German word embeddings learned by our model indeed satisfy the clustering constraint. Our model achieves competitive classification result with `para_sum`.

However, our model has not only the word embeddings but also the capability of computing the distributed representation of arbitrary word sequences, so we propose computing the document vectors in this manner. We call this method `para_doc`. We find that `para_doc` gives significantly better results than `para_sum`, especially on the `de`→`en` direction. In Table 1, we present our classification results on the CLDC task and compare them against strong baselines. Specifically, we first show the results of the original baselines in (Klementiev et al., 2012), then we show the stronger baselines (Chandar A P et al., 2014; Hermann and Blunsom, 2014), which perform considerably better but the gains are uneven between `en`→`de` and `de`→`en`. Finally, we show that our method works better than all these baselines. While `para_sum` outperforms the all the baselines but still with a significantly worse result in `de`→`en`, `para_doc` achieves better results and at once, avoids the asymmetry of all the other approaches.

Model	en→de	de→en
Majority class	46.8	46.8
Glossed	65.1	68.6
Machine translation	68.1	67.4
I-Matrix	77.6	71.1
Autoencoder	91.8	72.8
Compositional Add+	87.7	77.5
Compositional Bigram+	86.1	79.0
para_sum	90.6	78.8
para_doc	**92.7**	**91.5**

Table 1: Performances on CLDC English-German. Each model is trained on one language and tested on the other one. The numbers reported are the percentage of correctly predicted test documents. The first four baselines (Klementiev et al., 2012) are less sensitive to languages, so we do not observe large difference between the tasks `en`→`de` and `de`→`en`. Other methods that involve weighted sum of word vectors by (Chandar A P et al., 2014) and (Hermann and Blunsom, 2014) perform better on `en`→`de` than on `de`→`en`. Our work bridges the gap and simultaneously achieves state-of-the-art performance on both tasks.

5 Discussion

5.1 Symmetry of multilingual model

The key to succeed on the CLDC task is that equivalent documents in English and German should be mapped into similar points in the joint semantic space. This goal, however, is hard to achieve by using the idf weighted sum of word vectors in documents as proposed by (Klementiev et al., 2012). The major reason for this is perhaps due to the linguistic asymmetry between English and German. For example, verbs in German have more conjugations than their English counterparts and there is generally a large number of compound words in German. These phenomena are evidenced by the fact that the German vocab size is about twice of that of English ($95K$ versus $43K$) according to our training data. As a result, many German words appear less often, giving the model less opportunity to optimize for their representations. All these observations explain that it is inferior to simply represent documents as as weighted sum of the embeddings of their words. As highlighted in Table 1, methods that adopt the weighted average approach all suffer from the discrepancy between `en`→`de` and `de`→`en` CLDC results. Note that such asymmetry also holds for our learned word vectors when we simply sum them up (the `para_sum` row).

On the other hand, our second approach to compute document vectors does not suffer from this problem. At train time, we have already aimed at learning the same distributed representation for sentences (the clustering constraint on vectors of equivalent words follows only as a consequence). At test

time, the same clustering constraint leads the document vectors computed by our model to be more symmetric than the weighted sum of word vectors. This symmetrization explains why our classification results on en→de and de→en using para_doc are about equally strong, and both are better than those of para_sum.

5.2 Effects of embedding dimensions

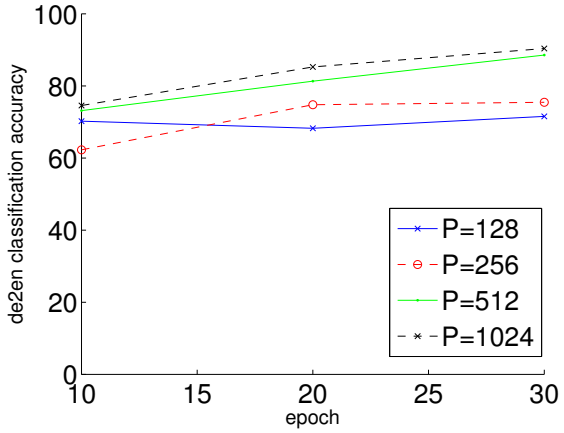

Figure 2: Test results for the de→en CLDC task across training epochs. Larger P gives better result while converging slower at test time.

We train four models on English-German data with with $D = 128$ and $P \in \{128, 256, 512, 1024\}$ and compare their test performances as training progresses. As demonstrated in Figure 2, models with larger P give better classification results (though they require more test iterations to converge to good sentence embeddings).

6 Conclusion

In summary, we have presented our novel approach to computing distributed representations of arbitrary word sequences in different languages from unannotated parallel data. Our method achieves state-of-the-art performance on a bilingual benchmark between English and German. We also gave our intuitions to explain why the model works even though it is nondeterministic while computing sentence vectors at test time. Further intuitions also suggest that it is possible to incorporate new languages into the model without hurting the previously known one. In the future, we plan to investigate the model's capacity to learn embeddings in other languages, such as French.

Acknowledgment

We gratefully acknowledge support from a gift from Bloomberg L.P. and from the Defense Advanced Research Projects Agency (DARPA) Broad Operational Language Translation (BOLT) program under contract HR0011-12-C-0015 through IBM. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of DARPA, or the US government. We also thank members of the Stanford NLP Group as well as the annonymous reviewers for their valuable comments and feedbacks.

References

Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *JMLR*, 3:1137–1155.

Sarath Chandar A P, Stanislas Lauly, Hugo Larochelle, Mitesh Khapra, Balaraman Ravindran, Vikas C Raykar, and Amrita Saha. 2014. An autoencoder approach to learning bilingual word representations. In *NIPS*.

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *EMNLP*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *ICML*.

Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR*, 12:2493–2537.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Stephan Gouws, Yoshua Bengio, and Greg Corrado. 2014. Bilbowa: Fast bilingual distributed representations without word alignments. In *NIPS Deep Learning Workshop*.

Karl Moritz Hermann and Phil Blunsom. 2014. Multilingual Models for Compositional Distributional Semantics. In *ACL*.

E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *ACL*.

Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. 2012. Inducing crosslingual distributed representations of words. In *COLING*.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *MT Summit*.

Tomáš Kočiský, Karl Moritz Hermann, and Phil Blunsom. 2014. Learning Bilingual Word Representations by Marginalizing Alignments. In *ACL*.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *ICML*.

Minh-Thang Luong, Richard Socher, and Christopher D. Manning. 2013. Better word representations with recursive neural networks for morphology. In *CoNLL*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *NAACL-HLT*.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Interspeech*.

Tomas Mikolov, Stefan Kombrink, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. In *ICASSP*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *ICLR*.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013b. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Andriy Mnih and Geoffrey Hinton. 2009. A scalable hierarchical distributed language model. In *NIPS*.

Frederic Morin. 2005. Hierarchical probabilistic neural network language model. In *AISTATS*.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013a. Parsing With Compositional Vector Grammars. In *ACL*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*.

Will Y. Zou, Richard Socher, Daniel Cer, and Christopher D. Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *EMNLP*.