Cross-Linguistic Phoneme Embeddings for Computational Historical and Typological Linguistics

Marlon Betz

July 6, 2016

Contents

1 Introduction

Embeddings nowadays build the backbone of every Deep Learning NLP architecture. Due to their capacity to encode a vast amount of latent semantic and syntactic information without the need of previous manual construction, they gave rise to the contemporary Deep Learning boom, i.e. deep neural networks that can capture hidden features in data sets and by that have allowed for huge performance gains in numerous NLP fields.
While embeddings are currently being used for several linguistic units such as characters [Kim et al.2015, dos Santos and Zadrozny2014, Zhang et al.2015], words [Mikolov et al.2013a,Mikolov et al.2013b,Pennington et al.2014] or entire sentences [Kiros et al.2015], proper phonemes have been largely been excluded from this trend, although there are some data that would indeed allow for it and research areas where they could be of great use, especially in the fields of Computational Typological and Historical Linguistics, which the

current deep learning boom has hardly touched yet. There, even though it is clear that some phonemes share more common features and such form natural classes, they are often treated as pure symbols that share a common distance between each other. Even phoneme representation models that do incorporate phonological features are often handcrafted [Kondrak2000,Rama2016] or include task-specific information that inherently suffer from restricted generalization abilities when used for other tasks [Jäger2014]. Moreover, those methods usually reduce the number of possible phonemes to a minimum, getting rid of important information such as secondary or co-articulations.
In this paper, I will first a discuss a theoretical motivation for using data-driven phoneme embeddings instead of plain symbolic representations or hand-crafted feature encodings. I will then shortly take a look on related research. A big part of this paper will then review several embedding models. I will then discuss intrinsic evaluation methods for the embeddings and use those to compare the performances of the previously described models. This is then followed by a discussion of several use cases that could be interesting for those interested in data-driven approaches to Typological and Historical Linguistics. Finally, I will recapitulate the benefits and drawbacks of phoneme embeddings in a final resume.

2 Theoretical and Practical Motivation

3 Related Research

Word embeddings have proven to be a reliable tool for several NLP tasks. However, finer grained embedding models that take care of the internal structure of a word have been shown to give similar or even improved performance over traditional word embeddings as they are able to encode morphological [dos Santos and Zadrozny2014] or semantic [Chen et al.2015] information even for out-of-vocabulary items [Ling et al.2015]. This follows the simple idea that since a word can be split up into its respective morphemes that encode morphosyntactic and semantic information, encoding subparts of a word instead of a the whole word should allow for better generalization for unseen words without the need to compute and store vector representations for every word type.

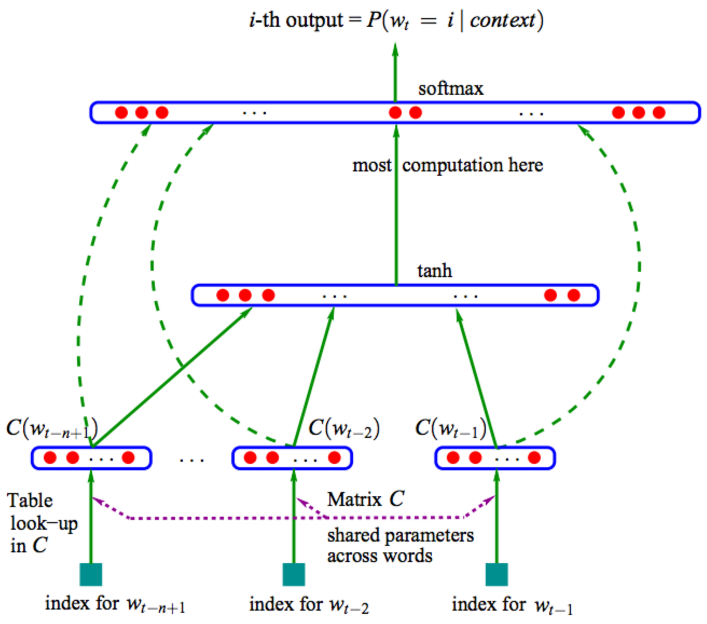

Figure 1: The neural language model as proposed in [Bengio et al.2003].

4 Embedding Models

4.1 Count-based Embedding Models

4.1.1 Latent Semantic Analysis

4.1.2 GloVe

4.1.3 SPPMI

4.1.4 SPPMI-SVD

4.2 Neural Embedding Models

Models that make use of cooccurrence count matrix factorizations usually suffer from the need of storing huge amounts of data into working memory. Neural networks with hidden layers, on the other hand, have been proven to be able to approximate any given hidden function while they still allow for online or mini-batch training. This is a huge improvement over models that only work with whole data batches, as much less working memory has to be used here. [Bengio et al.2003] pioneered as the first to introduce Neural Language Models in general. Here, a network with two hidden layers is proposed, where a first non-linear hidden layer with locally shared weights gives embeddings of single words in the given context window, while the next layer is a concatenation of the single word embeddings and another non-linear activation function and gives a context embedding. The last layer than is a softmax classifier that predicts the next word (cf. Figure 1). It later became clear that this model had two major drawbacks:

- The softmax classifier needs to compute the probability of all word types in the vocabulary. This is a major flaw also of following models (see below).
- The non-linear hidden layer severely worsened the convergence rate of the network.

[Collobert and Weston2008] then improved the model tremendously by getting rid of the expensive softmax classifier by introducing a score based loss function:

$$J_\theta = \sum_{x \in X} \sum_{w \in V} \max(0, 1 - f_\theta(x) + f_\theta(x^{(w)})) \tag{1}$$

Here, the the correct windows x containing n words are sampled from the set of all possible windows X in the corpus, while for every window x, a corrupted version $x^{(w)}$ is produced by replacing the center word of x by the another word of the vocabulary V. The objective then becomes maximizing the distance between the scores output by the model fore correct and the incorrect window with a margin of 1. The actual model then was still the same as in [Bengio et al.2003]. This model could already embed semantically similar words close to each other, but still suffered from the non-linear hidden layer, which slowed down learning enormously ([Bengio et al.2003] report a training time of 7 weeks for a vocabulary of 130000 word types)

4.2.1 Word2Vec

Among general-purpose embedding models in general and among neural embedding models in particular, the word2vec algorithm [Mikolov et al.2013a, Mikolov et al.2013b] stands out as major breakthrough, as it offers good performance while the parameters to train are less then with the previous models. Different to the previous models in [Bengio et al.2003, Collobert and Weston2008], which employ non-linear hidden layers, here the model only consists of a linear encoder layer that computes the actual embeddings. It can be seen as a full departure from traditional embedding models as a subgroup of language models towards a new kind of model family that in the first place tries to encode semantic and morphosyntactic information of a word rather than to predict the next word itself. Instead, the context window usually encompasses all surrounding words of a given target word instead of just the previous words. This makes it rather unhandy for true language modeling, but allows for much better incorporation of latent information. Moreover, the model can be trained in two ways: It either tries to predict the target word given its context (Continuous Bag-of-Words; CBOW) or tries to predict the context of a word given the target word itself

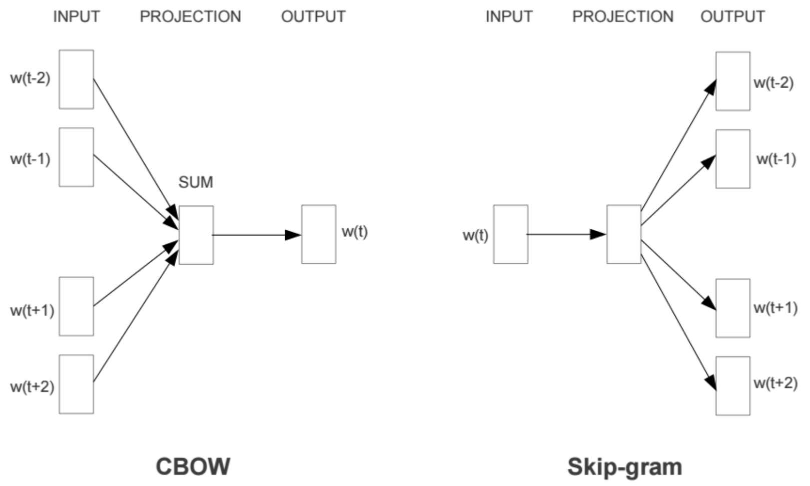

Figure 2: The two common architectures of word2vec. The Continuous Bag-of-Words (CBOW) model predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word. From [Mikolov et al.2013a].

(Skip-Gram, cf. Figure 2). The corresponding loss function for the CBOW model is given as

$$J_\theta = \frac{1}{T}\sum_{t=1}^{T} \log p(w_t \mid w_{t-n}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+n}) \tag{2}$$

where $p(w_t \mid w_{t-1}, \ldots, w_{t-n+1})$ is given as a softmax

$$p(w_t \mid w_{t-1}, \ldots, w_{t-n+1}) = \frac{\exp(h^\top v'_{w_t}))}{\sum_{w_i \in V} \exp(h^\top v'_{w_i})} \tag{3}$$

while the loss for the Skip-Gram model is given as

$$J_\theta = \frac{1}{T}\sum_{t=1}^{T} \sum_{-n \leq j \leq n, \neq 0} \log p(w_{t+j} \mid w_t) \tag{4}$$

where $p(w_{t+j} \mid w_t)$ is again defined as a softmax

$$\log p(w_{t+j} \mid w_t) = \frac{\exp(v_{w_t}^\top v'_{w_{t+j}}))}{\sum_{w_i \in V} \exp(h^\top v'_{w_i})} \tag{5}$$

where v_{wi} is an input vector and v'_{wi} an output vector representation of a word w_i and $h^\top$ the transpose of the sum of the output vectors of the words in the context. Again, the calculation of the softmax here includes summing over all contexts in the corpus, which computationally is expensive. Further research largely concentrated on finding a way to define p as a memory- and time-friendly approximation of the true softmax probability. A first solution was given by using a hierarchical softmax classifier [Morin and Bengio2005] nstead. Here, the classifier layer is a binary tree with the actual words as leaves. At every node, the network learns to either follow the left or the right branch with a certain probability as in Eq. 5 that is equal to the sum of the child elements of the respective branch, but is actually computed as the product of $h^\top$ and the output vector of the word at node n pulled through a logistic sigmoid:

$$p(right|n, c) = \sigma(h^\top v'_n) \tag{6}$$

This means in order to calculate the softmax probability of word, we only have to follow the path down to the word leaf instead of summing over all vocabulary entries. For binary trees, this means we only have to pass at most $\log_2(\mid V \mid)$ nodes to calculate the probability of a word, which is a huge performance boost over the traditional softmax classifier.

5 Evaluation

5.1 Data

5.2 Evaluation Methods

5.3 Results

6 Use Cases

6.1 Phonemic String Comparison

6.2 Modeling Sound Change

6.3 Phoneme Inventory Clustering

7 Resume

References

[Bengio et al.2003] Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *journal of machine learning research*, 3(Feb):1137–1155.

[Chen et al.2015] Chen, X., Xu, L., Liu, Z., Sun, M., and Luan, H. (2015). Joint learning of character and word embeddings. In *Proceedings of IJCAI*, pages 1236–1242.

[Collobert and Weston2008] Collobert, R. and Weston, J. (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. ACM.

[dos Santos and Zadrozny2014] dos Santos, C. N. and Zadrozny, B. (2014). Learning character-level representations for part-of-speech tagging. In *ICML*, pages 1818–1826.

[Jäger2014] Jäger, G. (2014). Phylogenetic inference from word lists using weighted alignment with empirically determined weights. In *Quantifying Language Dynamics*, pages 155–204. Brill.

[Kim et al.2015] Kim, Y., Jernite, Y., Sontag, D., and Rush, A. M. (2015). Character-aware neural language models. *arXiv preprint arXiv:1508.06615.*

[Kiros et al.2015] Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., and Fidler, S. (2015). Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302.

[Kondrak2000] Kondrak, G. (2000). A new algorithm for the alignment of phonetic sequences. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 288–295. Association for Computational Linguistics.

[Ling et al.2015] Ling, W., Luís, T., Marujo, L., Astudillo, R. F., Amir, S., Dyer, C., Black, A. W., and Trancoso, I. (2015). Finding function in form: Compositional character models for open vocabulary word representation. *arXiv preprint arXiv:1508.02096.*

[Mikolov et al.2013a] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781.*

[Mikolov et al.2013b] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

[Morin and Bengio2005] Morin, F. and Bengio, Y. (2005). Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, pages 246–252. Citeseer.

[Pennington et al.2014] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–43.

[Rama2016] Rama, T. (2016). Siamese convolutional networks based on phonetic features for cognate identification. *arXiv preprint arXiv:1605.05172.*

[Zhang et al.2015] Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.