ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

Unsupervised Word Representations Learning with Bilinear Convolutional Network on Characters

Thomas Luka[1], Laure Soulier[2] and David Picard[1] *

1- LIGM, École des Ponts, Univ Gustave Eiffel, CNRS
Marne-la-Vallée - France

2- Sorbonne Université, CNRS, LIP6
F-75005 Paris, France

Abstract. In this paper, we propose a new unsupervised method for learning word embedding with raw characters as input representations, bypassing the problems arising from the use of a dictionary. To achieve this purpose, we translate *the distributional hypothesis* into a unsupervised metric learning objective, which allows to consider only an encoder instead of an encoder-decoder architecture. We propose to use a convolutional neural network with bilinear product blocks and residual connections to encode co-occurrences patterns. We show the effectiveness of our approach by comparing it with classical word embedding methods such as fastText and GloVe on several benchmarks.

1 Introduction

Word representation learning is a key task in natural language processing since it is usually at the basis of methods tackling a wide variety of tasks including text retrieval, question answering or machine translation. Recent approaches focus on contextualized word representations (the word representation depends on the context so a word have multiple representations), like in [1], with the major drawback that a word cannot have a representation without its context. On the contrary, we focus in this work on uncontextualized word representations.

Recent methods for learning uncontextualized word representations use either a dictionary of the most frequent words [2], either sub-word units [3, 4], or character convolutions [5]. These methods present several drawbacks: for the dictionary ones, words which are not in the dictionary, like rare words and words with typographic errors, are all associated to the same unknown token and thus have the same representation whereas they don't share the same meaning. In addition, these methods usually require the training of a decoder which is discarded after training.

Our work tackles the problems arising from the use of a dictionary. To avoid a decoding step we reformulate the *distributional hypothesis* into a metric learning objective function which directly puts words with similar context closer in the embedding space. Furthermore, we take inspiration from computer vision where

*This work is co-funded by *Agence Innovation Défense* (AID) and École des Ponts ParisTech

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

bilinear pooling has shown significant improvements in several tasks including fine-grained recognition [6, 7, 8] and visual question answering [9]. The hypothesis is that bilinear pooling can be useful in convolutional word embeddings to capture co-occurrences between characters.

To that end, we propose BiCharaConv (**Bi**linear Pooling on **Chara**cter **Conv**olution), a convolutional word encoder which implements bilinear pooling operations trained using unsupervised metric learning. We demonstrate the effectiveness of our approach on several benchmarks and demonstrate the relevance of our method.

2 Related Work

In 2013, Mikolov et al [2] introduce the well-known CBOW and skip-gram methods to learn word representations. It first builds a dictionary of the N most frequent words and then project them in the embedding space.Later, Pennington et al introduce GloVe [10], a new model which links matrix factorization approach with methods based on a context-window like skip-gram using a global log-bilinear regression model. However, the drawbacks of these methods is that all words not included in the dictionary have the same representation. Worse, words with typographic errors are encoded to the unknown word. Besides, a decoder is trained whereas it is useless after training.

To tackle these problems, several options are proposed. Wieting et al [11] choose to use n-grams, sub-word units of fixed length. A character sequence is represented as the sum of its n-grams. This idea is also exploited by Bojanowski et al. [3] with fastText. Inspired by compression algorithms, other authors use Byte-Pair-Encoding (BPE) at a character level [4] to encode words (rare words can be then represented as a sequence of known sub-word units) when still other authors use character convolutions [5]. However, all these methods still need a decoder for training.

The most recent researches propose to include context information inside the word representations. Peters et al in [1] include features extracted thanks to convolutions on characters and all the intermediate states of a biLSTM inside the word representations in ELMo. More recently, models based on transformers [12] which implements a self-attention mechanism, like BERT [13] and its variants, achieve state-of-the-art on a wide variety of sentence analysis tasks. The major drawback of these methods is that they cannot compute the representation of a word without its context.

In this work, we focus on single word encoder (unlike ELMo or BERT) that avoids the limitations introduced by the use of a dictionary and by the encoder/decoder training scheme thanks to unsupervised metric learning. We evaluate how competitive can be a bilinear convolutional architecture.

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

3 Proposed Method

The main goal of our architecture is to detect co-occurrences with *OR* and *AND* logics. The logic *OR* allows to encode option like "there is a 'c' *OR* a 'd' at position p" and implements with 1D-convolutions. On the contrary, the logic *AND* allow to encode combinations, like "there is a 'c' at position p *AND* a 'l' at position $p+1$" and implements thanks to bilinear pooling operations described below. We hypothesize that the combination of these logics is able to detect patterns of characters like prefix, stem, or suffix robust to typos.

To perform the *AND* logic, we propose to encode particular combination of characters by using bilinear pooling to detect directions in the co-variance matrix of nearby features (like character features). More formally, let us consider the following features: $\mathbf{x_t}$ and k-shifted features $\mathbf{x_{t-k}}$. We want in fact to compute:

$$\begin{aligned}\sum_k \langle \mathbf{x_{t-k}}{\mathbf{x_t}}^T, \mathbf{W_k} \rangle &= \sum_k \left\langle \mathbf{x_{t-k}}{\mathbf{x_t}}^T, \sum_{i=1}^{r} \lambda_i \mathbf{u_{i,k}}{\mathbf{v_i}}^T \right\rangle \\ &= \sum_k \langle \Lambda, [\langle \mathbf{x_{t-k}}, \mathbf{u_{i,k}} \rangle \langle \mathbf{x_t}, \mathbf{v_i} \rangle]_i \rangle \ (\Lambda = [\lambda_i]_i) \end{aligned} \quad (1)$$

with $\mathbf{W_k}$ learned projections matrices with a low-rank assumption. In practice, the matrices $\mathbf{W_k}$ inspect the co-variance matrices of the features[1] to detect the directions representing a specific pattern due to a specific character combination. Equation 1 is easily implementable with an element-wise multiplication and 1D-convolutions.

In details, to implement logic *OR*, we use 1D-convolutions followed by a BatchNormalization and a LeakyReLU activation ($C^{kernel\_size}_{nb\_filters}$). To implement *AND* logic, we use the previously described (Eq. 1) bilinear pooling blocs (B). We combine this two logics into the following architecture:

$$\begin{aligned} X \rightarrow C^1_{512} \rightarrow \left[C^3_{512} \rightarrow C^1_{512}\right] \rightarrow B^5_{512} \rightarrow P^2 \\ \rightarrow \left[C^3_{512} \rightarrow C^1_{512}\right] \rightarrow B^3_{1024} \rightarrow FC_{300} \rightarrow z \end{aligned} \quad (2)$$

Here P^{window_size} denotes a MaxPooling operation, FC_{output_dim} denotes a fully connected layer, $[\cdot]$ denotes a residual connection, X is the input word representation and z is the representation in the embedding space.

X is a matrix $\in \mathcal{M}_{l_w \times d_c}(\mathbb{R})$ where each line x_j encodes the j^{th} character of a word of length l_w. We choose to encode only lowercase letters of alphabet in addition to special characters begin/end of word and unknown character. Characters are represented as a set of orthonormal vectors with dimension d_c.

To train our word encoder without a decoder, the idea is to convert the *distributional hypothesis* into a metric learning objective function. The goal is to put closer in the embedding space words which appear in the same sentence and to push away words which appear in different sentences. Thus, words with similar meaning, which appear in similar context, will be close in the embedding

[1] In practice, we compute the second order matrices but the reasoning is the same.

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

space by transitivity. To achieve this purpose we use a triplet loss [14]. The triplets are composed of a query word w_q randomly samples in the training set, a positive word randomly samples in the same sentence as w_q and a negative word randomly samples in a sentence different from the one of w_q.

Besides, to avoid a degenerate solution, we add an orthogonalization loss. The total objective function becomes:

$$\mathcal{L} = \mathcal{L}_{triplet} + \gamma \left[\frac{1}{N} \left(\sum_{w \in \mathcal{B}} z z^T \right) - \frac{1}{N} \sum_{w \in \mathcal{B}} z \frac{1}{N} \sum_{w \in \mathcal{B}} z^T - I \right]^2 \tag{3}$$

N is the number of words in a batch $\mathcal{B}$, z the representation of the word w in the embedding space and γ a coefficient to adjust. In fact we force the estimation of the batch co-variance matrix to be equal to the identity matrix (matrix I).

4 Results

After training our model on a corpus composed of a mix of subset of Wikipedia, Common-Crawl, UMBC webbase and MS-COCO sentences (around $47B$ tokens), we follow [15] and evaluate our method on several tasks which reflect our capacity to capture concreteness of a word and similarity/relatedness between two words and our capacity to predict features of a noun - like *is round* or *is edible* for the noun apple. For the concreteness experiments, the score reported is the coefficient of determination ($\mathcal{R}^2$) of a SVR with Gaussian kernel on our embedding with human scores as ground truth. For the similarity/relatedness experiments, the score is the mean of the Pearson correlation (σ) between the cosine similarity of two words in the embedding space and the similarity given by humans on several similarity benchmarks. For the features predictions the score reported is the mean on the 43 characteristics to predict of the F1-score between the predictions obtained with a linear SVM and the ground truth. We refer the reader to [15] for more details about evaluation methods and benchmarks.

We choose two classical and popular methods as a baseline: GloVe pretrained vectors and fastText pretrained vectors (obtained from the authors' website).

The table 1 shows that our method is as good as GloVe and fastText but with more homogeneous results on all the tests. Indeed, the two strengths of fastText are the ability to capture concreteness in the embedding and the ability to predict feature norms whereas its scores in similarity are lower. GloVe does well on similarity but have lower scores in the two other evaluation methods. Our method captures at the same time concreteness, similarity and feature norms.

5 Ablation study

To demonstrate the benefits of the Bilinear Pooling (B blocks), we train two models: one corresponding to a simpler architecture than the full model (with less convolution filters, LeakyReLU replaced by ReLU and FC_{300} replaced by

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

Test	GloVe	fastText	BiCharaConv
Concreteness ($\mathcal{R}^2$) ↑	0.68	**0.71**	0.70
Similarity (σ) ↑	**0.53**	0.45	0.52
Feature Norm (F1-Score) ↑	0.44	**0.52**	0.47
Mean ↑	0.55	**0.56**	**0.56**

Table 1: Comparison between GloVe, fastText and our model BiCharaConv

GAP i.e. GlobalAveragePooling). This architecture is represented below:

$$X \to C^1_{256} \to \left[C^3_{256} \to C^1_{256}\right] \to B^5_{256} \to P^2 \\ \to \left[C^3_{256} \to C^1_{256}\right] \to B^3_{256} \to GAP \to z \quad (4)$$

The other architecture is the same but with the B^k_f blocks replaced by C^k_f. On table 2, it appears clearly that the bilinear pooling gives a gain over all results. The effect is more visible on similarity (7% gain) and feature norm (9% gain) tests. On concreteness test, there is only 4% gain. These results demonstrate the usefulness of bilinear pooling in a character convolutional architecture.

	Concreteness	Similarity	Feature Norm	Mean
WithBP	**0.66**	**0.42**	**0.42**	**0.50**
WithoutBP	0.62	0.35	0.33	0.43

Table 2: Influence of the Bilinear Pooling

On figure 1, we inspect the influence of the orthogonalization coefficient γ on the embedding space. On the scores, we discover that the orthogonalization of features has an effect especially on the similarity benchmarks. Besides, it seems that high values of γ destruct the embedding even if the features are orthogonal

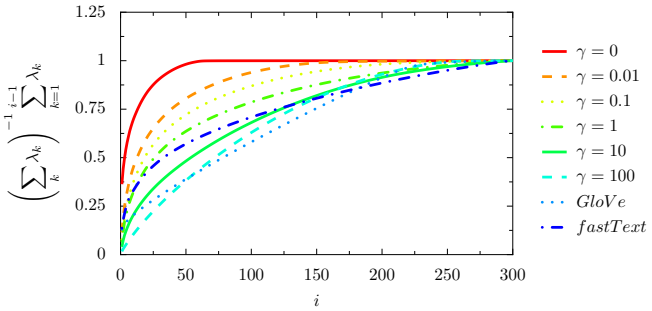

Fig. 1: Cumulative sum of the singular values of the covariance matrix of the embedding evaluation words features for BiCharaConv with different values for γ, for GloVe and for fastText. Best view in colors

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

whereas too small values have little or no effect. The better trade off between orthogonalization and embedding evaluation scores is obtained with $\gamma = 0.1$.

6 Conclusion

In this work we propose a novel approach to learn word representations with a character-based input. We use in our architecture BiCharaConv convolutions directly on characters, residual connections to ease training process and retain low level information and bilinear pooling to detect pattern of characters. Our method tackles issues of dictionary, n-grams and convolutional methods. First we resolve the problem of out-of-vocabulary words and typos. Second, by translating the *distributional hypothesis* into a metric learning objective, we avoid the learning of a decoder which is thrown after training. Our approach is as efficient as widely used methods. In future work, the use of bilinear pooling must be explored in *transformer* architectures to see if it can enhance the sentence representations learning.

References

[1] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. *NAACL*, 2018.

[2] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *ICLR Workshop*, 2013.

[3] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *TACL*, 5, 2017.

[4] R. Sennrich, B. Haddow, and A. Birch. Neural machine translation of rare words with subword units. In *ACL*, 2016.

[5] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush. Character-aware neural language models. In *AAAI*, 2016.

[6] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear CNN models for fine-grained visual recognition. In *ICCV*, 2015.

[7] Y. Gao, O. Beijbom, N. Zhang, and T. Darrell. Compact bilinear pooling. In *CVPR*, 2016.

[8] S. Kong and C. Fowlkes. Low-rank bilinear pooling for fine-grained classification. In *CVPR*, 2017.

[9] J. Kim, K. W. On, W. Lim, J. Kim, J. Ha, and B. Zhang. Hadamard product for low-rank bilinear pooling. In *ICLR*, 2017.

[10] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[11] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu. Charagram: Embedding words and sentences via character n-grams. In *EMNLP*, 2016.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *NIPS*, 2017.

[13] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[14] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *ICCV*, 2007.

[15] E. Zablocki, B. Piwowarski, L. Soulier, and P. Gallinari. Learning multi-modal word representation grounded in visual context. In *AAAI*, volume 32, 2018.