Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 108C (2017) 2343–2347

Procedia Computer Science

International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland

par2hier: towards vector representations for hierarchical content

Tommaso Teofili

Adobe Systems,
teofili@adobe.com

Abstract

Word embeddings have received a lot of attention in the natural language processing area for their capabilities of capturing inner words semantics (e.g. word2vec, GloVe). The need of catching semantics at a higher and more abstract level led to creation of models like paragraph vectors for sentences and documents, seq2vec for biological sequences. In this paper we illustrate an approach for creating vector representations for hierarchical content where each node in the hierarchy is represented as a (recursive) function of its paragraph vector and the hierarchical vectors of its child nodes, computed via matrix factorization. We evaluate the effectiveness of our solution against flat paragraph vectors on a text categorization task obtaining significant μF1 improvements.

Keywords: word embeddings, paragraph vectors, factorization methods, hierarchical content

1 Introduction

Learning vector representations of text is useful to solve a wide range of tasks, like clustering, text classification but also ranking in information retrieval. A common approach to create such vectors is the bag-of-words (BOW) where each document is represented as a vector whose elements are (eventually weighted) text term frequencies. BOW, despite its simplicity, has been widely used as it gave considerably good results; on the other hand it disregards term ordering and provides a sparse representation so that such a vector can have lots of zeros for all the words it doesn't contain. In recent years however research has focused mainly on dense models that do not suffer from the curse of dimensionality. Factorization methods like SVD but especially neural network based algorithms have shown good results in capturing semantics of words and documents when creating such vectors, also called embeddings. Such vectors have been used increasingly in NLP applications beyond just representation but also as underlying features to build models for named entity recognition, word sense disambiguation, parsing, tagging and machine translation. The need to capture higher abstractions of text entities has led to the definition of algorithms that can create embeddings for more than just words; some are specific for certain domains like sequence vectors for biological sequences, some others, like paragraph vectors, are more general and can be used in different contexts (sentences, paragraphs, documents, etc.). Hierarchically structured contents like documents with sections and subsections (e.g. research papers) can be modelled using paragraph vectors but miss to take the hierarchy into account while building the embeddings.

In this paper we extend paragraph vectors algorithm in order to apply it to hierarchical content and create embeddings that take into account the relationships among nodes in the hierarchy, so that the semantics conveyed by a certain piece of text in the hierarchy are influenced by its descendants whereas paragraph vectors can only provide a flattened representation. A key factor in our research is that we choose not to create yet another neural network model, but rather to use matrix factorization when computing a node's hierarchical vector to extract the topics from its child nodes. We find out that hierarchical embeddings built this way have a greater capability of capturing similarity of nodes/vectors that belong to the same (sub) hierarchy; consequently they seem to be more appropriate to be used in NLP tasks that apply to hierarchical contents, as evidenced in our experiments in text categorization tasks.

We make the following contributions: define an algorithm to build embeddings for hierarchical content (text) on top of paragraph vectors, demonstrate such hierarchical vectors overperform paragraph vectors on text categorization of hierarchical text.

2 Related work

In [10] two neural network models called SkipGram and Continuous Bag of Words were proposed in order to extract continuous vector representations of words; that work, also known as *word2vec*, and similar such approaches like GloVe [12], were subsequently used as building blocks for learning dense vector representations for word sequences, in particular the work on paragraph vectors [7], which we extend in this paper, is very flexible and allows to extract embeddings at a different granularity (e.g. sentences, paragraphs, documents). Models for extracting vector representations for vertical domains like biological sequences were also created, as in [6]. Several research efforts keep focusing on embeddings for words from different perspectives [16] [15]. Particularly relevant to our research is the work which provides a novel neural network model for learning hierarchical paragraph vectors [3].

3 Algorithms

We propose to create vector representation for documents whose contents are organized according to a hierarchy, composed by nodes; notable examples of such documents are books and articles. Our approach, based on the usage of paragraph vectors [7] as baseline vector representation, aims to capture the semantics implicitly brought by relationships in a hierarchy. We outline an algorithm for creating hierarchical embeddings where each node's embedding recursively depends on its descendants' embeddings.

3.1 Recursive smoothed paragraph vectors

Given a set of hierarchical documents we first learn each node's flat dense vector representation using the neural network approach called Paragraph Vectors outlined in [7] using PV-DM method for each *node* in the hierarchy as if they were flat documents, disregarding node interconnections at first. Then we bring back each node's links bottom up, from leaves to the top, according to the following recursive program:

1. on a leaf node: hiearchical vectors are simply paragraph vectors
2. on a non leaf node: the k centroids of the n child nodes of current node are used to *smooth* the node's paragraph vector

The *smoothing* task relies on k-means clustering; since it can be reduced to a matrix factorization problem, as in [17] [1], we decide to use Singular Value Decomposition [4] matrix factorization, in particular truncated SVD [5] approximation which allows to accumulate k singular values along both dimensions in a matrix. We purposely discard Negative Matrix Factorization, despite its good results for large matrices, because of its complexity [14] and also because in the case of text is uncommon to have high number of child nodes in a hierarchy. On the other hand SVD has shown good results for clustering [11], [8], produces a good rank-k approximation keeping an efficient representation of the relationship between data elements.

We inspect two different ways of *smoothing* a node's paragraph vector by its child nodes' vectors: one based on performing a two pass clustering and one based on a single pass clustering and a vector sum.

3.2 Two pass clustering smoothing

For each node, the first pass consists on performing k-means clustering, by applying truncated SVD algorithm, to the matrix of child nodes' hierarchical vectors. This way we approximate and reduce the dimensionality of the vectors and cluster the most important "topics" that should influence the current node embedding. In the second pass we perform another truncated SVD with k set to 1 in order to obtain a single final hierarchical vector.

A truncated Singular Value Decomposition $M = U_t \Sigma_t V_t^*$ is similar to a classic SVD $M = U \Sigma V^*$, except that only the t column vectors of U and t row vectors of V^* corresponding to the t largest singular values Σ_t are calculated. We treat matrix factorization as a relaxed k-means clustering and therefore we can apply t-SVD where t is the chosen number of clusters; consequently we choose the t vectors from V^* as the centroids because they keep the dimensionality of the rows of the original matrix.

Given a certain node h in a hierarchy having n children, we perform k-means clustering using t-SVD over the matrix composed using each child node (hierarchical) embedding and extract k vectors given by V^t. Such vectors represent the k most important topics brought by current node's children. We then compose a new matrix whose rows are the k topic vectors plus node h paragraph vector and perform another pass of k-means clustering using t-SVD with $k = 1$, the resulting V^t vector represents the hierarchical vector for node h.

3.3 Cluster and sum smoothing

For each node, the first pass is exactly the same described in the previous section, which performs k-means clustering by applying truncated SVD to the matrix of child nodes' hierarchical vectors, but it produces the final hierarchical vector by doing a vector sum between the node's paragraph vector and the k topic vectors.

4 Experiments

We evaluate our method for building hierarchical vectors against "plain" paragraph vectors in a text categorization task.

We created a dataset based on 5000 research papers extracted from Google Scholar and transformed into plain text using Apache Tika [9]. Each paper is organized in a hierarchy that corresponds to common research paper templates: $document > (abstract, section) > subsection > paragraph > subparagraph$ (we explicitly excluded references). We tag each node in the hierarchy according to such taxonomy and use K Nearest Neighbour [2] for performing the categorization task. For the sake of capturing how vector similarity changes with hierarchical vectors, given a set of documents D each composed by a number of paragraphs in P, we define an index of intra document similarity as a measure of the likelihood that a certain vector nearest neighbour belongs to the same hierarchy.

$$IIDS = \frac{1}{|D| \cdot |P|} \sum_{d \in D} \sum_{p \in d} |nearest(p) \in d| \tag{1}$$

We consider our results, implemented using Deeplearning4j [13], to be statistically significant with a p-value equal or less than 0.05.

We calculate paragraph vectors and hierarchical vectors for each abstract, section, subsection, paragraph. When using our approach we get an improvement in the described text categorization task up to 0.04 with hierarchical embeddings using *cluster and sum* method (*P2H-CS*) and up to 0.14 improvement when using hierarchical embeddings using *two pass cluster* method (*P2H-2PC*) when compared to "plain" paragraph vectors (*PV*).

Embeddings	k	μF1	IIDS
P2H-2PC	4	0.94	0.92
P2H-2PC	3	0.92	0.92
P2H-2PC	2	0.91	0.93
P2H-CS	4	0.84	0.93
P2H-CS	3	0.82	0.94
P2H-CS	2	0.83	0.95
PV		0.80	0.90

Table 1: Text categorization and IIDS results

We additionally notice an improvement of up to 0.03 in the IIDS with hierarchical embeddings using *two pass cluster* and an improvement up to 0.05 using *cluster and sum* methods.

Aggregating the results from the text categorization task and IIDS measurements, we can see that both our methods work better than plain paragraph vectors in finding similar text that belong to the same category in the hierarchical taxonomy, additionally they tend to locate vectors closer as they belong to the same document (higher IIDS). A simplified implementation of the algorithms and experiments we did can be found on Github [1].

5 Conclusions

We outline a method for learning dense vector representation of hierarchical content, based on the extraction of dense vector representations of single text nodes via paragraph vectors recursively weighted by using truncated SVD over child nodes. Our experiments show that such hierarchical vectors outperforms flat paragraph vectors on a text categorization task over hierarchical documents. Also we found out that hierarchical embeddings belonging to the same

[1]https://github.com/tteofili/par2hier

(sub) hierarchy tend to lie closer in the vector space when extracted using our approach and therefore they can more accurately represent the document they belong to.

Our experiments focus on categorization over hierarchical text extracted from research papers, however we expect that other similarly hierachically organized documents like books or web pages could benefit from such vector representation, additionally the presented techniques could be used on other areas of computer science like page clustering for information extraction from the web.

6 Acknowledgments

We would like to thank Federico Valeri, Antonio Sanso, Suneel Marthi for providing feedback that helped improving this paper.

References

[1] T. Blumensath. Directional clustering through matrix factorization. *IEEE Trans. Neural Netw. Learning Syst.*, 27(10):2095–2107, 2016.

[2] P. Cunningham and S. J. Delany. k-nearest neighbour classifiers. *Multiple Classifier Systems*, 34:1–17, 2007.

[3] L. Elmer. Hierarchical paragraph vectors. 2015.

[4] G. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis*, 2(2), 1965.

[5] P. C. Hansen. The truncatedsvd as a method for regularization. *BIT Numerical Mathematics*, 27(4):534–553, 1987.

[6] D. Kimothi, A. Soni, P. Biyani, and J. M. Hogan. Distributed representations for biological sequence analysis. *CoRR*, 2016.

[7] Q. V. Le and T. Mikolov. Distributed representations of sentences and documents. In *ICML*, volume 14, pages 1188–1196, 2014.

[8] S. Lee and M. H. Hayes. Properties of the singular value decomposition for efficient data clustering. *IEEE Signal Process. Lett.*, 11(11):862–866, 2004.

[9] C. Mattmann and J. Zitting. *Tika in action.* Manning Publications Co., 2011.

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

[11] A. Mirzal. Clustering and latent semantic indexing aspects of the singular value decomposition. *IJIDS*, 8(1):53–72, 2016.

[12] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[13] D. Team. Deeplearning4j: Open-source distributed deep learning for the jvm. *Apache Software Foundation License*, 2.

[14] S. A. Vavasis. On the complexity of nonnegative matrix factorization. *SIAM Journal on Optimization*, 20(3):1364–1377, 2009.

[15] A. K. Vijayakumar, R. Vedantam, and D. Parikh. Sound-word2vec: Learning word representations grounded in sounds. *CoRR*, 2017.

[16] L. Yang, X. Chen, Z. Liu, and M. Sun. Improving word representations with document labels. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(4):863–870, 2017.

[17] H. Zhao, Z. Ding, and Y. Fu. Multi-view clustering via deep matrix factorization. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 2921–2927, 2017.