ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2.
Available from http://www.i6doc.com/en/.

Tensor Decompositions in Deep Learning

Davide Bacciu[1], Danilo P. Mandic[2] *

[1] Department of Computer Science - University of Pisa
Largo Bruno Pontecorvo, 3, 56127, Pisa - Italy
[2] Department of Electrical and Electronic Engineering - Imperial College
Exhibition Road, London, United Kingdom

Abstract. The paper surveys the topic of tensor decompositions in modern machine learning applications. It focuses on three active research topics of significant relevance for the community. After a brief review of consolidated works on multi-way data analysis, we consider the use of tensor decompositions in compressing the parameter space of deep learning models. Lastly, we discuss how tensor methods can be leveraged to yield richer adaptive representations of complex data, including structured information. The paper concludes with a discussion on interesting open research challenges.

1 Introduction

In the latter years, tensor methods have been gaining increasing interest in the machine learning community. Multiway data analysis is one of their earliest and most popular application, addressing the processing of large scale and highly complex data, such as multivariate sensor signals. A tensor can be seen as a generalization of multidimensional arrays where the traditional algebraic operators are extended accordingly (e.g. element-wise sum, inner and outer products, etc). A tensor representation allows to capture complex interactions among input features which would not be evident on *flattened* data [1]. They are a flexible data structure allowing to seamlessly reshape vectorial data to a tensor representation for multi-linear analysis (tensorization) and viceversa (vectorization). Clearly, any analysis performed on a full tensorial representation easily results in so called curse-of-dimensionality problems, with the complexity growing exponentially with the tensor order. This is where tensor decompositions play a fundamental role, allowing to reduce the complexity of the representation space and preserving computational feasibility of the analysis, without a drastic reduction in the ability to capture high-order relationships in the data [1]. Tensor decompositions operate similarly to their matrix counterpart, by decomposing high-dimensional tensors into a sum of products of lower dimensional factors.

Apart from their direct application to multi-way input data analysis, tensors are widely adopted as a fundamental building block for machine learning models. Firstly, they have found application in a variety of machine learning paradigms, ranging from neural networks [2, 3] to probabilistic models [4], to enable the efficient compression of the model parameters leveraging tensor decomposition methods. Secondly, they provide a means to extend existing vectorial machine

*This work has been supported by MIUR under project SIR 2014 LIST-IT (RBSI14STDE).

learning models to capture richer data representations, where tensor decompositions provide the necessary theoretical and methodological backbone to study, characterize and control the expressiveness of the model [5].

This tutorial paper takes pace from the homonym special session of the *28th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning* to provide a focused survey of the use of tensor decompositions in deep learning models and applications. Given the constraints of a short communication, we will not provide a detailed introduction to tensor methods, whereas we will focus on reviewing three interesting and broad research topics, following the themes of the contributions presented in the session. An interested reader will find in literature comprehensive survey works introducing tensor decompositions [6], their applications to data analysis [5] and machine learning [7]. The remainder of the paper is organized as follows: Section 2 provides a brief background on tensors and notable decompositions. Section 3 reviews *classical* applications of tensor decompositions to multi-way signal processing and data analysis. Section 4 discusses more works leveraging decompositions in neural model compression, while Section 5 focuses on the relevant topic of tensor methods as enabler for learning more expressive representations of complex data, including structured samples such as trees, networks and graphs. The paper concludes with a discussion of interesting open challenges in the field.

2 Tensor decomposition overview

In the following, we provide a brief introduction to tensors and their terminology. We also present three popular tensor decompositions that are representative of three classes of methods characterized by different assumptions and aimed to capture different types of relationships in multi-way data. For a more in-depth introductions readers are referred to [6] and [1].

2.1 Tensors: definition and terminology

A tensor describes a multilinear relationship between algebraic objects which span from a scalars to tensors themselves. Its definition can be given based on different mathematical concepts. For the sake of simplicity, we introduce tensors based on their intuitive view as a a multi-dimensional array. In this context, a tensor $\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is an element of the tensor product of d vector spaces, such that the corresponding multi-dimensional array is $\mathbf{A}(i_1, \ldots, i_d) \in \mathbb{R}$, $i_k \in [1, n_k]$ where:

- i_k represents the index along the k-th dimension, known as *mode*;
- d is the *order* of $\mathbf{A}$, i.e. the number of modes;
- n_k represents the *size* along the k-th mode.

Subarrays can be extracted from the full tensor by fixing a subset of the i_k indices. Notable subsarrays are the *fibers*, which are the multi-way equivalent of vectors and are defined by fixing every index but one, and the *slices*, that are defined by fixing all but two indices. The *norm* of a tensor is the square root of the sum of the squares of all its elements, i.e. analogous to a matrix Frobenius norm. The *inner product* of two same-sized tensors is the sum of the element-wise products of their entries. The *tensor product* is the generalization of the vector outer product concept to tensors, i.e. through the outer product of d vectors we can obtain the tensor $\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$:

$$\mathbf{A} = \mathbf{a}_1 \odot \mathbf{a}_2 \odot \cdots \odot \mathbf{a}_d$$

where $\mathbf{a}_k$ is the n_k-dimensional vector corresponding to the k-th mode in the tensor. A relevant concept is that of *rank-1* tensor, that is a d-way object which can be strictly decomposed as the tensor product of d vectors. A tensor decomposition expresses a tensor in terms of a sequence of sum and products operating on simpler multi-way objects. In the following we summarize three relevant and popular methods heavily inspired by matrix decomposition.

2.2 Canonical Polyadic (CP) decomposition

The CP decomposition [8] is a representative of the family of rank decompositions, that seek to express a tensor as the sum of a finite number of rank-1 tensors. There are several analogous formulations for the CP (see [6] for a detailed account), the most general being

$$\mathbf{A} = \sum_{r=1}^{R} \lambda_r \mathbf{a}_1(r) \odot \mathbf{a}_2(r) \odot \cdots \odot \mathbf{a}_d(r) \tag{1}$$

where $\mathbf{a}_1(r) \odot \mathbf{a}_2(r) \odot \cdots \odot \mathbf{a}_d(r)$ is the r-th rank-1 tensor and $\lambda_r \in \mathbb{R}^R$. The value of r is called *canonical rank*.

2.3 Tucker decomposition (TD)

The TD [9] is a representative of the family of Higher-Order Singular Value Decompositions (HOSVD), that seek to find the components that best capture the variation in single modes independently one-another. The TD decomposes a tensor into a *core tensor*, defining how the different tensor items interact and are mixed with each other, and multiple *modes matrices*. Given a tensor $\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, the TD of its $(i_1, \ldots, i_d)$ entry is

$$\mathbf{A}(i_1, \ldots, i_d) = \sum_{r_1=1}^{R_1} \cdots \sum_{r_d=1}^{R_d} \mathbf{G}(r_1, \ldots, r_d) U_1(i_1, r_1) \ldots U_d(i_d, r_d), \tag{2}$$

where $\mathbf{G} \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ is the core tensor and $U_k \in \mathbb{R}^{n_k \times r_k}$ are the mode matrices. The values R_k denote the rank along the k-th dimension.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

2.4 Tensor Train (TT) decomposition

The TT [10] is a representative of those decompositions assuming that mode ordering is somehow semantically relevant and should be taken into consideration in the factorization. The TT approximates a tensor $\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ through a sequence of order-3 tensors, each connected to its left and right neighbor in the mode ordering. More formally, the TT of the $(i_1, \ldots, i_d)$ entry of $\mathbf{A}$ is

$$\mathbf{A}(i_1, \ldots, i_d) = \sum_{r_0=1}^{R_0} \cdots \sum_{r_d=1}^{R_d} \mathbf{G}_1(r_0, i_1, r_1)\mathbf{G}_2(r_1, i_2, r_2) \ldots \mathbf{G}_d(r_{d-1}, i_d, r_d), \quad (3)$$

where $\mathbf{G}_k \in \mathbb{R}^{R_{k-1} \times n_k \times R_k}$ are tensors. The value R_k represents the TT-rank along the k-th mode and it is such that $R_0 = R_d = 1$.

3 Multi-way data analysis

Among the different uses of tensor decompositions in machine learning, multi-way data analysis has certainly been the first to develop and, yet, the most popular [1]. By multi-way analysis, we refer to the fact that tensor decompositions are used as the adaptive method to process, organize and make sense of the available data and its complex relationships, without resorting to *external* learning models. In this brief survey, we focus on those data analysis applications which, for complexity and scale, are typically within the scope of deep learning applications.

A type of data relationship that is often addressed through tensor decompositions is the sequential one, i.e. where the data collection comprises samples which follow a complete order. A typical example is timeseries data: this is typically a 3-way tensor in which slices represent a snapshot of the data at a given time. Here, tensor decomposition allows to isolate latent structures in the data: [11], for instance, leverages CP to discover patterns in the temporal evolution of publishing activities and to perform look-ahead predictions, while in [12] CP is used for detecting anomalies.

One of the works presented in the special session falls into this line of research. In [13], the authors discuss an application of tensor decomposition on short text game commentary data to understand the temporal changes in the effectiveness of cricket players. To this end, the paper shows the use of a Tucker decomposition for discrete data. The paper also releases the original data collected and used for the analysis.

Tensor decompositions can be applied to more general forms of relational data. For instance, samples characterized by the presence of different types of relationship (e.g. friendship, content sharing, tagging in social networks) can be straightforwardly tensorized by modeling each relationship in terms of the two subjects involved and the type or relation binding them, having a different tensor slice to represent each of the available relations. In [14], for instance, it is presented a natural language understanding application, where tensors are

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

used to capture multiplicative interactions combining predicate, object and subject generating aggregated representations for event prediction tasks. In [15], a Tucker-like decomposition is exploited on three-way tensors to perform collective learning on network data. The use of tensor methods in graph data processing appears a lively research field, in particular as regards the prediction of new relations in knowledge graphs, where tensor decomposition show their advantages in the trade-off between expressivity and computational efficiency [16, 17].

Another successful application of multi-way analysis is in information fusion: for instance, [18] uses tensors to fuse visual and textual representations. Overall, the applications discussed in this section seem to share a general limitation, that is related with the use of tensors with a fixed number of modes, that rarely exceeds three.

4 Neural model compression

Tensor decompositions have a second major application to machine learning which is related to their compression ability, in particular when considering the typical large-scale of data in deep learning. The relationship of tensors with the deep learning world are particularly evident when considering *tensor networks* [5] that provide a general framework for representing the factorization of large tensors into networks of smaller elements. Tensor networks provide the means to effectively regulate the trade-off between the number of model parameters and the predictive accuracy. Further, they put forward a methodological framework that allows assessing the expressive power of the compressed neural models. The typical approach to compress whole deep learning architectures is discussed in detail by [5]. In brief, given an uncompressed deep neural network (DNN), one can construct the corresponding tensor network representation. This can be then simplified, by the decompositions discussed in Section 2, to achieve the desired trade-off between parameterization and predictive accuracy. Finally, the compressed tensor network is mapped back into the corresponding compressed DNN. This approach has been successfully applied to Restricted Boltzmann Machines [19] and convolutional architectures [20, 21].

A different approach to neural model compression leverages tensor decomposition on the single layers of the network. For instance, [3] proposes to efficiently store the dense weight matrices of the fully-connected layers of a VGG network by leveraging Tensor Train factorization. Conversely, [22] introduces the Tucker Tensor Layer as an alternative to the dense weight-matrices of neural networks. They show how to leverage tensor decomposition to drastically reduce the number of parameters in the neural layer, also deriving a specialized form of back-propagation on tensors that preserves the physical interpretability of Tucker decomposition and provides an insight into the learning process of the layer.

An alternative perspective over the use of tensor factorization in neural layers is provided by [23], which introduces a multitask representation learning framework leveraging tensor factorisation to share knowledge across tasks in fully

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

connected and convolutional DNN layers.

5 Higher-Order Representation Learning

Our brief overview concludes with the most recent and, possibly, yet less developed research topic binding deep learning models and tensor decompositions. This topic mixes the application of multi-way analysis to relational data discussed in Section 3 with the use of tensors as elements of the neural layer reviewed in Section 4. In particular, we consider how tensors can be used to modify input aggregation functions inside the artificial neuron. Input aggregation in neurons is typically achieved by a weighted sum of the inputs to the unit which, for vectorial data, is equivalent to the inner product between a the weight vector and the input vector. When dealing with tree structured data, this process in generalized in such a way that, given a specific node in the tree, the neuron recursively computes its activation by a weighted sum of the activations of its children, with appropriate weight sharing assumptions. Such an aggregation function can be easily tensorized, as postulated already by [24], to capture higher-order relationships between the children encodings.

The higher-order recursive neuron by [24] has long been only a theoretical model, formulated solely for binary trees. In [25, 26, 27], the model has been extended to a probabilistic formulation for general n-ary trees which clearly highlights the tensorial nature of the input aggregation function (that is a n-way tensor map). Nonetheless, for computational tractability issues, the same works approximated the tensor with a simple probabilistic factorization largely equivalent to a weighted sum in neural models. Only recently, [4] has introduced a proper tensor decomposition of the n-way probabilistic tensor leveraging a Bayesian Tucker decomposition.

On the side of neural models, [28] discusses the use of a high-order neural network for structured data that leverages a full 3-way tensor for aggregating children information in binary parse trees within a natural language processing application. The second paper [29] in this special session, instead, introduces what is seemingly the first tensor recursive neurons for n-ary trees. More importantly, it proposes the use of tensor decompositions as a viable and expressive trade-off between the simplicity of sum aggregation and the complexity of full tensor aggregation. Two input accumulation functions are discussed in the paper: one leveraging CP decomposition and the other relying on the Tensor-Train factorization. Similarly, [30] shows how a tensor recursive neuron can exploit the Tucker decomposition to control the trade-off between the size of the neural representation and the size of the neural parameter space.

6 Conclusions and Research Challenges

The use of tensor decompositions as a fundamental building block in deep learning models is a growing research topic which is, yet, in its infancy. Throughout our brief review, we have highlighted that the use of tensor factorization as a stand-alone method for multi-way data analysis is seemingly the most mature research topic. At the same time, tensor decompositions are starting to be heavily used for compression purposes in neural models, throughout well-grounded approaches that allow to compress full networks while maintaining performance guarantees, or by tensorization of full neural layers. Lastly, we have singled out a very recent and promising research direction leveraging tensor decompositions as input aggregation functions for building higher-order neurons for structured data processing, to learn more expressive neural representations of structured information.

The research themes discussed above stimulate interesting research challenges which can help increasing the effectiveness of deep learning models and deepen our understanding of their inner workings. Tensorized neural layers put forward a research question about whether tensorization should only affect the forward phase of neural models (i.e. computing of the neural activation) or if it has non-trivial reflections also on the backward phase (i.e. learning). In this respect, [22] began to discuss how Tucker decomposition can be leveraged in the backward phase to enhance interpretability of propagated gradients. Along these lines, it would be interesting to study if tensors can be leveraged to define novel weight optimization algorithms and whether they can contribute to the discussion concerning dynamics and convergence of stochastic gradient descent [31]. A second research challenge relates to representation learning with topology varying graphs. Literature reports wide use of tensors for multirelational data analysis on single networks and, as discussed in our review, some early works dealing with neural processing of collections of tree structured samples. What seems yet missing is their use in Deep Graph Networks (DGN) [32, 33] that can handle the most general case of datasets of graph samples with unconstrained topology. It seems likely that tensorization of the graph neurons would increase their ability in capturing higher-order structural dependencies between the nodes in the graphs. This might have a very practical impact on the predictive performance of the DGN which, as shown in [34], appears often lower than that of dummy baseline models. On the other hand, it would also be interesting to study how tensorization can affect the expressivity of DGN from a theoretical perspective [35, 36].

A key aspect to promote research on tensor decomposition in deep learning is the availability of software libraries integrating tensor methods within the deep learning development frameworks. While there are several consolidated libraries providing stable implementations of tensor decompositions, these are typically distributed as packages of popular scientific computing environments, such as R, Matlab and Mathematica, which are less used by the deep learning community. Nonetheless some contributions are beginning to appear also on this respect: `scikit-tensor` [37] integrates some consolidated tensor decomposition (Tucker, CP and the like) into the `scikit-learn` universe. More recently, `TensorD` [38] provides a Python tensor library built on `Tensorflow` and providing basic tensor operations and decompositions with support for parallel computation (e.g. GPU). `TensorLy` [39] is a Python library implementing a wide range of methods for tensor learning, allowing to leverage different computation back-ends including `NumPy`, `MXNet`, `PyTorch`, `TensorFlow`, and `CuPy`. `HOTTBOX` [40] is a recent standalone Python toolbox for tensor decompositions, statistical analysis, visualisation, feature extraction, regression and non-linear classification of multi-dimensional data.

References

[1] Andrzej Cichocki, Danilo Mandic, Lieven De Lathauwer, Guoxu Zhou, Qibin Zhao, Cesar Caiafa, and Huy Anh Phan. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine*, 32(2):145–163, 2015.

[2] Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan V. Oseledets, and Victor S. Lempitsky. Speeding-up convolutional neural networks using fine-tuned cp-decomposition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[3] Alexander Novikov, Dmitrii Podoprikhin, Anton Osokin, and Dmitry P Vetrov. Tensorizing neural networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 442–450. Curran Associates, Inc., 2015.

[4] Daniele Castellana and Davide Bacciu. Bayesian Tensor Factorisation for Bottom-up Hidden Tree Markov Models. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 7 2019.

[5] Andrzej Cichocki, Anh Huy Phan, Qibin Zhao, Namgil Lee, Ivan Oseledets, Masashi Sugiyama, and Danilo Mandic. Tensor networks for dimensionality reduction and large-scale optimizations: Part 2 applications and future perspectives. *Foundations and Trends in Machine Learning*, 9(6):431–673, 2017.

[6] Tamara G. Kolda and Brett W. Bader. Tensor Decompositions and Applications. *SIAM Review*, 51(3):455–500, 2009.

[7] Y. Ji, Q. Wang, X. Li, and J. Liu. A survey on tensor techniques and applications in machine learning. *IEEE Access*, 7:162950–162990, 2019.

[8] Henk A. L. Kiers. Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics*, 14(3):105–122, 2000.

[9] Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[10] Ivan V. Oseledets. Tensor-Train Decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 1 2011.

[11] Daniel M. Dunlavy, Tamara G. Kolda, and Evrim Acar. Temporal link prediction using matrix and tensor factorizations. *ACM Trans. Knowl. Discov. Data*, 5(2), February 2011.

[12] Evangelos Papalexakis, Konstantinos Pelechrinis, and Christos Faloutsos. Spotting misbehaviors in location-based social networks using tensors. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 551–552, New York, NY, USA, 2014. Association for Computing Machinery.

[13] Swarup Ranjan Behera and Vijaya Saradhi. Mining temporal changes in strengths and weaknesses of cricket players using tensor decomposition. In *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN'20)*, 2020.

[14] Noah Weber, Niranjan Balasubramanian, and Nathanael Chambers. Event representations with tensor-based compositions. In *AAAI*, 2017.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

[15] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 809–816, Madison, WI, USA, 2011. Omnipress.

[16] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing yago: Scalable machine learning for linked data. In *Proceedings of the 21st International Conference on World Wide Web*, WWW'12, pages 271–280, New York, NY, USA, 2012. Association for Computing Machinery.

[17] A. Padia, K. Kalpakis, and T. Finin. Inferring relations in knowledge graphs with tensor decompositions. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 4020–4022, Dec 2016.

[18] Hedi Ben-Younes, Rémi Cadene, Matthieu Cord, and Nicolas Thome. Mutan: Multimodal tucker fusion for visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2612–2620, 2017.

[19] Jing Chen, Song Cheng, Haidong Xie, Lei Wang, and Tao Xiang. Equivalence of restricted boltzmann machines and tensor network states. *Phys. Rev. B*, 97, Feb 2018.

[20] Nadav Cohen and Amnon Shashua. Convolutional rectifier networks as generalized tensor decompositions. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 955–963, New York, New York, USA, 20–22 Jun 2016. PMLR.

[21] Nadav Cohen and Amnon Shashua. Inductive bias of deep convolutional networks through pooling geometry. In *Proceedings of The International Conference on Learning Representations (ICLR 2017)*, 2017.

[22] Giuseppe Giovanni Calvi, Ahmad Moniri, Mahmoud Mahfouz, Zeyang Yu, Qibin Zhao, and Danilo P. Mandic. Tucker tensor layer in fully connected neural networks. *CoRR*, abs/1903.06133, 2019.

[23] Yongxin Yang and Timothy M. Hospedales. Deep multi-task representation learning: A tensor factorisation approach. In *International Conference on Learning Representations (ICLR)*, 2017.

[24] Paolo Frasconi, Marco Gori, and Alessandro Sperduti. A general framework for adaptive processing of data structures. *IEEE Transactions on Neural Networks*, 9(5):768–786, 1998.

[25] Davide Bacciu, Alessio Micheli, and Alessandro Sperduti. Compositional Generative Mapping for Tree-Structured Data - Part I: Bottom-Up Probabilistic Modeling of Trees. *IEEE Transactions on Neural Networks and Learning Systems*, 23(12):1987–2002, 2012.

[26] Davide Bacciu, Alessio Micheli, and Alessandro Sperduti. An input-output hidden Markov model for tree transductions. *Neurocomputing*, 112:34–46, 2013.

[27] Davide Bacciu, Alessio Micheli, and Alessandro Sperduti. Compositional Generative Mapping for Tree-Structured Data - Part II: Topographic Projection Model. *IEEE Transactions on Neural Networks and Learning Systems*, 2013.

[28] Richard Socher, Alex Perelygin, and Jy Wu. Recursive deep models for semantic compositionality over a sentiment treebank (RNTN). *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, 2013.

[29] Daniele Castellana and Davide Bacciu. Tensor decompositions in recursive neural networks for tree-structured data. In *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN'20)*, 2020.

[30] Daniele Castellana and Davide Bacciu. Generalising recursive neural models by tensor decomposition. Submitted, 2020.

[31] P. Chaudhari and S. Soatto. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–10, Feb 2018.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2.
Available from http://www.i6doc.com/en/.

[32] Davide Bacciu, Federico Errica, Alessio Micheli, and Marco Podda. A gentle introduction to deep learning for graphs, 2019.

[33] Davide Bacciu, Federico Errica, and Alessio Micheli. Contextual graph Markov model: A deep and generative approach to graph processing. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 294–303. PMLR, 2018.

[34] Federico Errica, Marco Podda, Davide Bacciu, and Alessio Micheli. A fair comparison of graph neural networks for graph classification. In *Proceedings of the Eighth International Conference on Learning Representations (ICLR 2020)*, Apr 2020.

[35] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *International Conference on Learning Representations (ICLR)*, 2019.

[36] Federico Errica, Davide Bacciu, and Alessio Micheli. Theoretically expressive and edge-aware graph learning. In Michel Verleysen, editor, *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN'20)*, Apr 2020.

[37] Maximilian Nickel and Evert Rol. `scikit-tensor` - a Python module for multilinear algebra and tensor factorizations. pypi.org/project/scikit-tensor-py3/, 2016.

[38] Liyang Hao, Siqi Liang, Jinmian Ye, and Zenglin Xu. Tensord: A tensor decomposition library in tensorflow. *Neurocomputing*, 318:196 – 200, 2018.

[39] Jean Kossaifi, Yannis Panagakis, Anima Anandkumar, and Maja Pantic. Tensorly: Tensor learning in python. *Journal of Machine Learning Research*, 20(26):1–6, 2019.

[40] Ilya Kisil, Giuseppe G. Calvi, Bruno Scalzo Dees, and Danilo P. Mandic. HOTTBOX: Higher Order Tensors ToolBOX. *Under review*, 2020.