A Gentle Introduction to Deep Learning for Graphs

Davide Bacciu[a], Federico Errica[a], Alessio Micheli[a], Marco Podda[a]

[a]*Department of Computer Science, University of Pisa, Italy.*

Abstract

The adaptive processing of graph data is a long-standing research topic that has been lately consolidated as a theme of major interest in the deep learning community. The snap increase in the amount and breadth of related research has come at the price of little systematization of knowledge and attention to earlier literature. This work is a tutorial introduction to the field of deep learning for graphs. It favors a consistent and progressive presentation of the main concepts and architectural aspects over an exposition of the most recent literature, for which the reader is referred to available surveys. The paper takes a top-down view of the problem, introducing a generalized formulation of graph representation learning based on a local and iterative approach to structured information processing. Moreover, it introduces the basic building blocks that can be combined to design novel and effective neural models for graphs. We complement the methodological exposition with a discussion of interesting research challenges and applications in the field.

Keywords: deep learning for graphs, graph neural networks, learning for structured data

1. Introduction

Graphs are a powerful tool to represent data that is produced by a variety of artificial and natural processes. A graph has a compositional nature, being a compound of atomic information pieces, and a relational nature, as the links defining its structure denote relationships between the linked entities. Also, graphs allow us to represent a multitude of associations through link orientation

and labels, such as discrete relationship types, chemical properties, and strength of the molecular bonds.

But most importantly, graphs are ubiquitous. In chemistry and material sciences, they represent the molecular structure of a compound, protein interaction and drug interaction networks, biological and bio-chemical associations. In social sciences, networks are widely used to represent people's relationships, whereas they model complex buying behaviors in recommender systems.

The richness of such data, together with the increasing availability of large repositories, has motivated a recent surge in interest in deep learning models that process graphs in an adaptive fashion. The methodological and practical challenges related to such an overarching goal are several. First, learning models for graphs should be able to cater for samples that can vary in size and topology. In addition to that, information about node identity and ordering across multiple samples is rarely available. Also, graphs are discrete objects, which poses restrictions to differentiability. Moreover, their combinatorial nature hampers the application of exhaustive search methods. Lastly, the most general classes of graphs allow the presence of loops, which are a source of complexity when it comes to message passing and node visits. In other words, dealing with graph data brings in unparalleled challenges, in terms of expressiveness and computational complexity, when compared to learning with vectorial data. Hence, this is an excellent development and testing field for novel neural network methodologies.

Despite the recent burst of excitement of the deep learning community, the area of neural networks for graphs has a long-standing and consolidated history, rooting in the early nineties with seminal works on Recursive Neural Networks for tree structured data (see [114, 41, 10] and the references therein). Such an approach has later been rediscovered within the context of natural language processing applications [113, 115]. Also, it has been progressively extended to more complex and richer structures, starting from directed acyclic graphs [89], for which universal approximation guarantees have been given [56]. The recursive processing of structures has also been leveraged by probabilistic approaches, first

as a purely theoretical model [41] and later more practically through efficient approximated distributions [6].

The recursive models share the idea of a (neural) state transition system that traverses the structure to compute its embedding. The main issue in extending such approaches to general graphs (cyclic/acyclic, directed/undirected) was the processing of cycles. Indeed, the *mutual dependencies* between state variables cannot be easily modeled by the recursive neural units. The earliest models to tackle this problem have been the Graph Neural Network [104] and the Neural Network for Graphs [88]. The former is based on a state transition system similar to the recursive neural networks, but it allows cycles in the state computation within a contractive setting of the dynamical system. The Neural Network for Graphs, instead, exploits the idea that mutual dependencies can be managed by leveraging the representations from previous layers in the architecture. This way, the model breaks the recursive dependencies in the graph cycles with a multi-layered architecture. Both models have pioneered the field by laying down the foundations of two of the main approaches for graph processing, namely the *recurrent* [104] and the *feedforward* [88] ones. In particular, the latter has now become the predominant approach, under the umbrella term of graph convolutional (neural) networks (named after approaches [72, 54] which reintroduced the above concepts around 2015).

This paper takes pace from this historical perspective to provide a gentle introduction to the field of neural networks for graphs, also referred to as deep learning for graphs in modern terminology. It is intended to be a paper of tutorial nature, favoring a well-founded, consistent, and progressive opening to the main concepts and building blocks to assemble deep architectures for graphs. Therefore, it does not aim at being an exposition of the most recently published works on the topic. The motivations for such a tutorial approach are multifaceted. On the one hand, the surge of recent works on deep learning for graphs has come at the price of a certain forgetfulness, if not lack of appropriate referencing, of pioneering and consolidated works. As a consequence, there is the risk of running through a wave of rediscovery of known results and models.

ploy, we can partition most deep graph learning models into *recurrent*, *feedforward* and *constructive* approaches. We now discuss how they work and what their differences are.

Recurrent Architectures. This family of models implements the iterative processing of node information as a dynamical system. Two of the most popular representatives of this family are the Graph Neural Network [104] and the Graph Echo State Network [44]. Both approaches rely on imposing contractive dynamics to ensure convergence of the iterative process. While the former enforces such constraints in the (supervised) loss function, the latter inherits convergence from the contractivity of (untrained) reservoir dynamics. The Gated Graph Neural Network [78] is another example of recurrent architecture where, differently from [104], the number of iterations is fixed a priori, regardless of whether convergence is reached or not. An iterative approach based on *collective inference*, which does not rely on any particular convergence criteria, was introduced in [83].
This family of models handles graph cycles by modeling the mutual dependencies between node states using a single layer of recurrent units. In this case, we can interpret the symbol ℓ of Figure 4 as an "iteration step" of the recurrent state transition function computed for the state of each node. Finally, we mention the recent Fast and Deep Graph Neural Network [45], a multi-layered and efficient version of the Graph Echo State Network.

Feedforward Architectures. In contrast to recurrent models, feedforward models do not exploit an iterative diffusion mechanism over the same layer of recurrent units. Instead, they stack multiple layers to *compose* the local context learned at each step. As a result, the mutual dependencies induced by cycles are managed via differently parameterized layers without the need for constraints to ensure the convergence of the encoding process. To draw a parallel with Figure 4 (here ℓ corresponds to the index of a layer), this compositionality affects the context of each node, which increases as a function of the network depth up to the inclusion of the entire graph [88]. The Neural Network for Graphs [88] was

where w_{c_k} is a learnable scalar parameter that weighs the contribution of arcs with label $\mathbf{a}_{uv} = c_k$, and $*$ multiplies every component of its first argument by w_{c_k}. This formulation presents an inner aggregation among neighbors sharing the same arc label, plus an outer weighted sum over each possible arc label. This way, the contribution of each arc label is learned separately. The Neural Network for Graphs [88] and the Relational Graph Convolutional Network [105] implement Eq. 7 explicitly, whereas the Contextual Graph Markov Model [3] uses the switching-parent approximation [103] to achieve the same goal. A more general solution, which works with continuous arc labels, is to reformulate Eq. 2 as

$$\mathbf{h}_v^{\ell+1} = \phi^{\ell+1}\Big(\mathbf{h}_v^{\ell},\ \Psi(\{e^{\ell+1}(\mathbf{a}_{uv})^T \psi^{\ell+1}(\mathbf{h}_u^{\ell}) \mid u \in \mathcal{N}_v\})\Big), \tag{8}$$

where e can be any function. Note how we explicitly introduce a dependence on the arc $\mathbf{a}_{uv}$ *inside* the neighborhood aggregation: this has the effect of weighting the contribution of each neighbor based on its (possibly multidimensional) arc label, regardless of whether it is continuous or discrete. For example, in [48] e is implemented as a neural network that outputs a weight matrix.

Attention. Attention mechanisms [119] assign a relevance score to each part of the input of a neural layer, and they have gained popularity in language-related tasks. When the input is graph-structured, we can apply attention to the aggregation function. This results in a weighted average of the neighbors where individual weights are a function of node v and its neighbor $u \in \mathcal{N}_v$. More formally, we extend the convolution of Eq. 2 in the following way:

$$\mathbf{h}_v^{\ell+1} = \phi^{\ell+1}\Big(\mathbf{h}_v^{\ell},\ \Psi(\{\alpha_{uv}^{\ell+1} * \psi^{\ell+1}(\mathbf{h}_u^{\ell}) \mid u \in \mathcal{N}_v\})\Big), \tag{9}$$

where $\alpha_{uv}^{\ell+1} \in \mathbb{R}$ is the *attention score* associated with $u \in \mathcal{N}_v$. In general, this score is unrelated to the edge information, and as such edge processing and attention are two quite distinct techniques. As a matter of fact, the Graph Attention Network [120] applies attention to its neighbors but it does not take into account edge information. To calculate the attention scores, the model

of the 2nd Workshop on Abusive Language Online (ALW2), pages 1–10, 2018.

[92] Federico Monti, Michael M. Bronstein, and Xavier Bresson. Geometric matrix completion with recurrent multi-graph neural networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 3700–3710, 2017.

[93] Galileo Mark Namata, Ben London, Lise Getoor, and Bert Huang. Query-driven active surveying for collective classification. In *Proceedings of the Workshop on Mining and Learning with Graphs*, 2012.

[94] Yaroslav Nechaev, Francesco Corcoglioniti, and Claudio Giuliano. SocialLink: exploiting graph embeddings to link DBpedia entities to Twitter profiles. *Progress in Artificial Intelligence*, 7(4):251–272, 2018. Publisher: Springer.

[95] Michel Neuhaus and Horst Bunke. Self-organizing maps for learning the edit costs in graph matching. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 35(3):503–514, 2005.

[96] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 701–710. ACM, 2014.

[97] Meng Qu, Yoshua Bengio, and Jian Tang. GMNN: Graph Markov Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 5241–5250, 2019.

[98] Liva Ralaivola, Sanjay J Swamidass, Hiroto Saigo, and Pierre Baldi. Graph kernels for chemical informatics. *Neural Networks*, 18(8):1093–1110, 2005. Publisher: Elsevier.

[99] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. struc2vec: Learning node representations from structural identity. In *Pro-*

ceedings of the 23rd International Conference on Knowledge Discovery and Data Mining (SIGKDD), pages 385–394. ACM, 2017.

[100] Veeru Sadhanala, Yu-Xiang Wang, and Ryan Tibshirani. Graph sparsification approaches for laplacian smoothing. In *Artificial Intelligence and Statistics*, pages 1250–1259, 2016.

[101] Bidisha Samanta, Abir De, Gourhari Jana, Pratim Kumar Chattaraj, Niloy Ganguly, and Manuel Gomez Rodriguez. NeVAE: A deep generative model for molecular graphs. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI)*, pages 1110–1117, 2019.

[102] Tae San Kim, Won Kyung Lee, and So Young Sohn. Graph convolutional network approach applied to predict hourly bike-sharing demands considering spatial, temporal, and global effects. *PloS one*, 14(9), 2019. Publisher: Public Library of Science.

[103] Lawrence K Saul and Michael I Jordan. Mixed memory Markov models: Decomposing complex stochastic processes as mixtures of simpler ones. *Machine Learning*, 37(1):75–87, 1999. Publisher: Springer.

[104] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009. Publisher: IEEE.

[105] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *Proceedings of the 15th European Semantic Web Conference (ESWC)*, pages 593–607. Springer, 2018.

[106] Ida Schomburg, Antje Chang, Christian Ebeling, Marion Gremse, Christian Heldt, Gregor Huhn, and Dietmar Schomburg. BRENDA, the enzyme database: updates and major new developments. *Nucleic Acids Research*, 32(suppl_1), 2004.

[107] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[108] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *Workshop on Relational Representation Learning, Neural Information Processing Systems (NeurIPS)*, 2018.

[109] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(Sep):2539–2561, 2011.

[110] Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. Efficient graphlet kernels for large graph comparison. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 488–495, 2009.

[111] Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3693–3702, 2017.

[112] Martin Simonovsky and Nikos Komodakis. GraphVAE: Towards generation of small graphs using variational autoencoders. In *Proceedings of the 27th International Conference on Artificial Neural Networks (ICANN)*, pages 412–422, 2018.

[113] Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 129–136, 2011.

[114] Alessandro Sperduti and Antonina Starita. Supervised neural networks for

[144] Ziwei Zhang, Peng Cui, and Wenwu Zhu. Deep learning on graphs: A survey. *CoRR*, abs/1812.04202, 2018.

[145] Zizhao Zhang, Haojie Lin, Yue Gao, and KLISS BNRist. Dynamic hypergraph structure learning. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3162–3169, 2018.

[146] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. In *Proceedings of the 21st Conferece on Neural Information Processing Systems (NIPS)*, pages 1601–1608, 2007.

[147] Marinka Zitnik, Monica Agrawal, and Jure Leskovec. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 34(13):i457–i466, 2018.

[148] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 2847–2856. ACM, 2018.