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/.

Theoretically Expressive and Edge-aware Graph Learning

Federico Errica[1], Davide Bacciu[1], Alessio Micheli[1]

[1] University of Pisa - Department of Computer Science
Largo Bruno Pontecorvo, 3, 56127, Pisa - Italy

Abstract. We propose a new Graph Neural Network that combines recent advancements in the field. We give theoretical contributions by proving that the model is strictly more general than the Graph Isomorphism Network and the Gated Graph Neural Network, as it can approximate the same functions and deal with arbitrary edge values. Then, we show how a single node information can flow through the graph unchanged.

1 Introduction

Graph Neural Networks (GNNs) [1, 2] have gained popularity as an efficient tool to process graph-structured data. The core idea underlying these models is the iterative aggregation of neighboring information to produce node representations. GNNs usually serve to solve node classification and graph classification tasks [3]. In recent years, researchers have proposed many architectures that mainly differ in the way neighborhood aggregation (also known as graph convolution) is performed. Some GNNs are proven to be able to discriminate the same graphs as the Weisfeiler-Lehman (WL) test of graph isomorphism [4, 5], while others focus on modeling edge labels [6, 7] and the recurrence in node representations [8]. In this work, we put these building blocks together to formalize a new family of GNNs that can handle arbitrary edges as well as the history of nodes and edges representations across layers. Our contributions are theorical: first, we show that the proposed network is strictly more expressive than the models it borrows from; then, we give further insights about contextual information spreading that add to those of [1].

2 Related Works

Two are the GNNs that inspired this work. The Graph Isomorphism Network (GIN) [4] is capable of discriminating the same structures as the 1-dim WL-test of graph isomorphism, and its architecture is fairly simple and efficient. Instead, the Gated Graph Neural Network (GG-NN) [8] is designed to take into account the history of node representations across the layers of the architecture, whereas the aggregation function is not backed up with theoretical results. While there are no formal guarantees about the expressiveness of GG-NN, the inductive bias imposed by the Gated Recurrent Unit (GRU) [9] incorporated in the graph convolution allows node representations to seamlessly flow across *layers*. This results in a neighborhood aggregation scheme that combines heterogeneous local

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/.

"views" of the graph. Finally, we mention that very few models for graph-structured data incorporate edge information in the learning process [6, 7], which is probably due to the fact that there are no common benchmarking datasets that contain attributed edge information. Nonetheless, the architecture we are about to define provides a theoretically more general tool to learn from graphs, as it combines the inductive bias of popular GNNs in a sound way.

3 Model

We now introduce our model, called Gated-GIN. We start by giving some notations; then, we present the details of the model.

Notation A graph $\mathbf{g} = (\mathcal{V}_g, \mathcal{E}_g, \mathcal{X}_g, \mathcal{A}_g)$ is formally defined by a set of nodes $\mathcal{V}_g$ and by a set of edges $\mathcal{E}_g$ between two vertices. Each node u is associated with a vector $x_u \in \mathcal{X}_g$. A directed edge (u,v) between nodes u and v is represented by a vector $a_{uv} \in \mathcal{A}_g$. The neighborhood of a node $u \in \mathcal{V}_g$ is defined as $\mathcal{N}(u) = \{v \in \mathcal{V}_g | (v,u) \in \mathcal{E}_g\}$, that is the set of nodes associated to incoming edges. We denote an hidden representation of a node v with h_v and that of an edge (u,v) with h_{uv}. Finally, we speak of *context* when a node's response depends on the information flowing through the structure, and we refer to the term "expressive" to say that a network is capable of approximating a certain family of functions.

3.1 Definition

Here, we extend the convolution of GIN [4] to deal with arbitrary edge values. Moreover, we incorporate the information propagation mechanism of GG-NN [8] to exploit the history of a node hidden representations across layers, rather than at different time steps.

Node convolution We start by defining the operations on attributed nodes at each layer k:

$$
\begin{aligned}
h_v^0 &= \phi_V^0(x_v), \\
z_v^k &= \sigma(\mathbf{W}_z^V[(1+\epsilon_V^k)h_v^{k-1}, \sum_{u \in \mathcal{N}(v)} h_u^{k-1} \odot h_{uv}^{k-1}] + \mathbf{b}_z^V), \\
r_v^k &= \sigma(\mathbf{W}_r^V[(1+\epsilon_V^k)h_v^{k-1}, \sum_{u \in \mathcal{N}(v)} h_u^{k-1} \odot h_{uv}^{k-1}] + \mathbf{b}_r^V), \\
\tilde{h}_v^k &= \phi_V^k((1+\epsilon_V^k)h_v^{k-1} \odot r_v^k + \sum_{u \in \mathcal{N}(v)} h_u^{k-1} \odot h_{uv}^{k-1}), \\
h_v^k &= (1-z_v^k) \odot h_v^{k-1} + z_v^k \odot \tilde{h}_v^k,
\end{aligned}
$$

where h^k is the hidden state, $\epsilon_V^k \in \mathbb{R}$ represents a learnable parameter, square brackets denote concatenation and $\odot$ the Hadamard product, σ is a gated activation function such as the sigmoid, ϕ is a multi layer perceptron (MLP), the

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/.

symbol $\mathbf{W}$ denotes a linear weight matrix and $\mathbf{b}$ its associated bias. The definitions of z and r are taken from GG-NN [8], which, in turn, was inspired by the gating functions of GRU [9]. Indeed, z and r represent the *update* and *reset* gate, respectively.

Edge convolution Similarly, we define edge representations at layer k using node and edge representations computed at layer $k-1$:

$$
\begin{aligned}
h_{uv}^0 &= \phi_E^0(a_{uv}),\\
z_{uv}^k &= \sigma(\mathbf{W}_E^z([h_{uv}^{k-1}, h_u^{k-1}, h_v^{k-1}]) + \mathbf{b}_z^E),\\
r_{uv}^k &= \sigma(\mathbf{W}_E^r([h_{uv}^{k-1}, h_u^{k-1}, h_v^{k-1}]) + \mathbf{b}_r^E),\\
\tilde{h}_{uv}^k &= \phi_E^k([h_{uv}^{k-1} \odot r_{uv}^k, h_u^{k-1}, h_v^{k-1}]),\\
h_{uv}^k &= (1 - z_{uv}^k) \odot h_{uv}^{k-1} + z_{uv}^k \odot \tilde{h}_{uv}^k.
\end{aligned}
$$

For undirected graphs, we can sum the contributions of h_u^{k-1} and h_v^{k-1} rather than concatenating them, so that $z_{uv} = z_{vu}$ and $r_{uv} = r_{vu}$. Usually, the recurrent architecture of GRU shares parameters across time steps to deal with sequences of variable length. In our case, a "time step" refers to one of the layers used to construct the architecture, hence the use of weight sharing is at the discretion of the user. In the rest of the paper, we assume a weight sharing technique, but the theoretical analysis of Section 4 is easily extendible.

4 Theoretical Analysis

This Section is devoted to provide a theoretical analysis of the proposed model. We start by proving that the method is at least as expressive as GIN [4], which implies it can discriminate the same structures as the 1-dim WL test.

Theorem 1. *Given a graph g and a node $v \in \mathcal{V}_g$, let $h_v^k = GIN^k(g) \in \mathbb{R}^d$ and $\hat{h}_v^k = \text{Gated-}GIN^k(g) \in \mathbb{R}^d$ be the outputs of the k-th graph convolution layer of GIN and Gated-GIN, respectively. Let us further assume that the multiset of neighboring states is countable. Then, for any choice of parameters θ_{GIN} of a GIN architecture with K layers, there exists a choice of parameters $\theta_{Gated-GIN}$ of a Gated-GIN architecture with K layers such that, for each $0 \leq k \leq K-1$ and $\epsilon > 0$, $||h_v^k - \hat{h}_v^k|| < \epsilon$.*

Proof. We proceed by induction. The statement trivially holds for $k = 0$; indeed, node representations can be generated using the same MLP. We now assume the statement holds for $k-1$, and we will prove that it holds for $k \leq K$ as well. First, we ignore the presence of edges by setting $h_{uv}^{k-1} = 1$. This can be done by choosing the parameters of the MLP associated with ϕ_E^k to represent the constant function $\phi_E^k(x) = \mathbf{1}$. It follows that we have $\tilde{h}_{uv}^k = \mathbf{1} \quad \forall (u,v) \in \mathcal{E}_g$. Secondly, we need to ignore previous node representations, that is $h_v^k = \tilde{h}_v^k$.

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/.

To obtain this, it is sufficient that $z_v^k = 1$ and $r_v^k = 1$; this holds in the limit when $\mathbf{W}_z^V \to \mathbf{0}, \mathbf{W}_r^V \to \mathbf{0}, \mathbf{b}_z^V \to +\infty$ and $\mathbf{b}_r^V \to +\infty$, resulting in

$$\lim_{\substack{b_r^V, b_z^V \to +\infty \\ W_r^V, W_z^V \to \mathbf{0}}} \hat{h}_v^k = \phi_V^k((1+\epsilon_V)h_v^{k-1} + \sum_{u \in \mathcal{N}(v)} h_u^{k-1}) = h_v^k.$$

□

Note that this proof is nearly identical when using MLPs instead of linear functions for the update and reset gates, as there is just one more matrix to consider. Moreover, we follow [4] and focus on the case where input node features belong to a countable set, which is not restrictive in practice.
The following Theorem is analogous to the previous one but for GG-NN. Before going on, we informally define a *multiset* as the set that allows for multiple instances for each of its elements.

Theorem 2. *Given a graph g and a node $v \in \mathcal{V}_g$, let $h_v^k = GG\text{-}NN^k(g) \in \mathbb{R}^d$ and $\hat{h}_v^k = Gated\text{-}GIN^k(g) \in \mathbb{R}^d$ be the outputs of the k-th graph convolution layer of GG-NN and Gated-GIN, respectively. Let us further assume that the multiset of neighboring states is countable. Then, for any choice of parameters θ_{GG-NN} of a GG-NN architecture with K layers, there exists a choice of parameters $\theta_{Gated-GIN}$ of a Gated-GIN architecture with K layers such that, for each $0 \leq k \leq K-1$ and $\epsilon > 0$, $||h_v^k - \hat{h}_v^k|| < \epsilon$.*

Proof. We again proceed by induction. For k=0, we recall that $h_v^0 = [x_v, \mathbf{0}]$, which can be obtained by a linear mapping Wx_v where W is a block matrix made by the identity matrix and the null matrix. Therefore, it follows from the universal approximation theorem [10] that $\hat{h}_v^0 = \phi_V^0(x_v)$ can approximate h_v^0.
If we assume that for each $0 < k \leq K$ and $\epsilon > 0$, $||h_v^{k-1} - \hat{h}_v^{k-1}|| < \epsilon$ and we use the same argument as in Theorem 1 to ignore edge labels, the inductive step follows from Lemma 5 of [4], i.e. Gated-GIN can approximate any function defined on multisets. □

In this work, we are not interested in studying the relation between GIN and GG-NN, as we have provided an architecture that is capable of approximating both. The next corollary, however, states that Gated-GIN is strictly more general than both GIN and GG-NN, as it can also handle edge attributes.

Corollary 1. *The class of functions of Gated-GIN is strictly larger than those of GIN and GG-NN.*

Proof. We will prove the statement for GIN, but the proof is identical for Gated-GIN. Let $\mathcal{F}_{GIN}$ and $\mathcal{F}_{Gated-GIN}$ the set of functions that GIN and Gated-GIN can approximate, respectively. It follows from Theorem 1 that $\mathcal{F}_{GIN} \subseteq \mathcal{F}_{Gated-GIN}$. Recall that, for any given graph g, $f \in \mathcal{F}_{\theta_{GIN}}$ ignores the contribution given by $\mathcal{A}_g$. Therefore, $\mathcal{F}_{GIN}$ corresponds to the set of functions such that $h_{uv}^k = 1 \ \ \forall k, (u,v) \in \mathcal{E}_g$. We conclude by saying that we can trivially construct a function $g \in \mathcal{F}_{Gated-GIN}$ such that $h_{uv}^k = 0 \ \ \forall k, (u,v) \in \mathcal{E}_g$, hence $\mathcal{F}_{GIN} \subset \mathcal{F}_{Gated-GIN}$. □

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/.

Despite these results about the ability of GNNs to discriminate certain structures, little is known about the requirements needed to effectively spread information across the graph. In the following, we study what is needed for GNNs to diffuse a single node information across the graph.

4.1 On context spreading of a single node

The formal analysis of the context provided in [1] characterizes how all nodes spread information across a graph. Indeed, using a deep GNN with k layers corresponds to making two nodes at distance k (indirectly) exchange their information. Here, we show that some GNNs can, in theory, spread a *single* node representation h_v^k across the graph without altering its value. Note that this result only applies to GNNs that compute a parametrized weighted sum of neighbors.

Theorem 3. *Given a graph g and a node $v \in \mathcal{V}_g$, assume we want to propagate an arbitrary $h_v^k \neq \mathbf{0}$ to node u at distance d such that $h_u^{k'} = h_v^k$, $k' > k$. Then there exists a permutation invariant function on a multi-set X of the form $g(X) = \phi(\sum_{x \in X} f(x))$ that can be approximated by the neighborhood aggregation of GNNs such that $k' = k + d$.*

Proof. The proof relies on the fact that an aggregation function that can make h_v^k seamlessly flow through the graph is $g(X) = \frac{1}{Z} \sum_{x \in X} \delta_{x,h_v^k} * x$, where δ_{x,h_v^k} is the Kronecker delta and $Z = \sum_{x \in X} \delta_{x,h_v^k}$ is a normalization term. This function is capable of ignoring values different from h_v^k, and takes an average when more than one value equal to h_v^k appears in the multiset. In summary, $g(X) = h_v^k$ if and only if $h_v^k \in X$, and 0 otherwise. By using this argument with the result of Theorem 2 in [1], it follows that there exists a $k' = k + d$ that satisfies $h_u^{k'} = h_v^k$. However, δ_{x,h_v^k} is a discontinuous function; as such, we need to show that it can be approximated by a continuous function, which in turn can be approximated by a neural network for [10]. To see this, consider the continuous function $f_{n,h_v^k}(h) = n^{-||h - h_v^k||_2}$; it is easy to show that the family of functions $\{f_{n,h_v^k}(h)\}$, $n > 1$ is *pointwise convergent* to δ_{h,h_v^k}. We conclude by saying that $f_{n,h_v^k}(h)$ can be approximated by an MLP with learnable parameter n. □

From a practical point of view, it may be very difficult to approximate $\delta_{h_v^k}(x)$ without imposing a more explicit inductive bias on the aggregation function. If the task at hand requires to move a node's information far away in the graph, one possibility is therefore to use the function $f_{n,h_v^k}(h) = n^{-||h,h_v^k||_2}$ to approximate $\delta_{h_v^k}$. Indeed, Theorem 3 assumes h_v^k is fixed, but we can treat it as a learnable parameter as well.

5 Conclusions

We have proposed a new architecture for GNNs that combines the inductive bias of the theoretically expressive Graph Isomorphism Network and the recurrent

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/.

mechanism of the Gated Graph Neural Network. We proved that the architecture does not lose expressivity with respect to both GNNs, which means one can now combine all the benefits together with no compromise. Moreover, we incorporate edge convolutions to deal with arbitrary edge attributes. As a result, the new network is strictly more expressive than those considered in this work. Finally, we give a sufficient requirement for GNNs to spread a single node representation across the graph, which is of practical importance in applicative contexts. Future works include the empirical application of such an architecture to new benchmarks where edge information is crucial to solve a task.

References

[1] Alessio Micheli. Neural network for graphs: A contextual constructive approach. *IEEE Trans. Neural Networks*, 20(3):498–511, 2009.

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

[3] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinícius Flores Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Çaglar Gülçehre, H. Francis Song, Andrew J. Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey R. Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matthew Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *CoRR*, abs/1806.01261, 2018.

[4] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[5] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 4602–4609, 2019.

[6] 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, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 304–313, 2018.

[7] Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 29–38, 2017.

[8] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. Gated graph sequence neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[9] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar*, pages 1724–1734, 2014.

[10] George Cybenko. Approximation by superpositions of a sigmoidal function. *MCSS*, 5(4):455, 1992.