Advanced Topics in Distributed Systems
Paper Review

Lodovico Giaretta

April 7, 2020

1 Introduction

The amount of data being collected, in various domains, with the goal to train Machine Learning models, is increasing rapidly. A large portion of this data does not consist of isolated data points. Instead, it includes relationships between entities, which may be an important aspect of the behaviour to be modeled. This has led, in recent years, to a growing interest in the field of Graph Representation Learning (GRL), a family of techniques that allows the relationships between entities to be embedded in a low-dimensional space and fed to traditional Machine Learning pipelines.

In particular, Graph Neural Networks (GNNs) are a family of GRL techniques based on Deep Learning, and they are able to produce embeddings that encapsulate both entity features and relationships. Furthermore, these embeddings can be fine-tuned for specific tasks. In this report, three papers are reviewed, representing three different approaches to the construction of GNNs. Furthermore, each of these papers has had a strong impact on the field, often affecting also other domains.

Papers:

- Li et al., *Gated Graph Sequence Neural Networks*, 2016
- Kipf et al., *Semi-Supervised Classification with Graph Convolutional Networks*, 2017
- Hamilton et al., *Inductive Representation Learning on Large Graphs*, 2017

2 Gated Graph Sequence Neural Networks

2.1 Problem Statement

The previous state-of-the-art in the area on Recurrent Graph Neural Networks (RGNNs), established by Scarselli et al. [2], predated the "Deep Learning era", was not based on modern Deep Learning techniques and was quite limited and inefficient. Furthermore, Li et al. identify the need for GRL techniques that can output sequences of embeddings, instead of a single set of embeddings for classification purposes.

2.2 Main Contributions

This paper introduces two main contributions:

- the introduction of Gated Graph Neural Networks (GG-NNs), an adaptation of the RGNN model by Scarselli et al. to modern Deep Learning standards, with higher efficiency and the ability to model a wider set of propagation functions;
- the introduction of Gated Graph Sequence Neural Networks (GGS-NNs), a model that takes a graph as input and can produce a sequence of embeddings as outputs.

The authors of the paper seem to focus more on the second contribution, with the first being a mere requirement in order to achieve the second. However, to the knowledge of the reviewer, not much research has followed on their footsteps with regards to outputting sequence from graphs. However, the introduction of modern RGNNs has been fundamental for the development of GRL, finding applications in areas such as computational chemistry and allowing subsequent work to unify different types of GNNs in a single framework.

2.3 Solution

To achieve their first contribution, Li et al. replace the generic propagation function in Scarselli et al. with a trainable Gated Recurrent Unit (GRU), and consequently replace the Almeida-Pineda training algorithm with Backpropagation Through Time (BTT). Furthermore, they allow the nodes to have initial annotations, which are then propagated by the GG-NN, which was not possible with the algorithm from Scarselli et al.

To achieve their second contribution, that is, to be able to produce sequence outputs, they train two GG-NNs. The first is tasked with producing a new graph at every iteration, starting from its own output in the previous iteration (or from the original graph at the beginning). The second GG-NN uses the current graph produced by the first to construct the actual output embeddings.

2.4 Strong Points

- By employing modern Deep Learning techniques, GG-NNs provide several advantage over the previous state-of-the-art: 1) they do not need to run to convergence, and thus do not require their weights to be constrained to guarantee that, allowing them to encode a wider range of transformations; 2) they only need to propagate for a small number of iterations, making training much faster and 3) they can employ initial annotations to mark nodes of interest and thus solve node-specific tasks.

- To the knowledge of the reviewer, GGS-NNs are the only models specifically designed to produce sequence outputs from graphs.

2.5 Weak Points

- The work on sequence-based GRL did not receive particular follow-up in the research community. The most likely reason is the lack of very compelling use-cases for such technology. Indeed, even the examples used in this paper do not seem particularly suited to the proposed architecture, which seems to be applied in a particularly convoluted way in order to, as it does not map to the problems in a straightforward way.

- While the formulation of RGNNs using modern recurrent units provides many advantages (as already described), it also causes a drawback: these recurrent units need to be trained via BTT, which requires intermediate activations and gradients to be stored, causing a large memory overhead. On the other hand, the original solution by Scarselli et al., based on running the GNN to full convergence, has a closed form that only involves the initial and final states.

3 Semi-Supervised Classification with GCNs

3.1 Problem Statement

Spectral graph convolutions were originally introduced by Hammond et al. [1] as a tool to apply trainable convolutional filters (as defined in the field of signal processing) to graph-structured data. However, the computation on these filters is a very demanding operation, which scales badly to large graphs. Subsequent works have reduced the complexity and increased the locality of these filters, but not to a sufficient extent.

3.2 Main Contributions

This paper introduces a fast, first-order approximation for a graph convolutional layer. Multiple such layers can be stacked together to build a deep Graph Convolutional Network (GCN) that is both powerful and efficient.

3.3 Solution

Previous approximations of graph convolutional filters were based on K-order Chebyshev polynomials, that can approximate the graph convolution operation using the normalized Laplacian directly, without the need to perform its eigen-decomposition.

Kipf et al. build on top of this previous work, by limiting K to 1, and further reducing the degrees of freedom by reducing the number of parameters in the layer. Furthermore, they apply various transformations, approximations and renormalization tricks in order to achieve high accuracy, low computational cost and to avoid overfitting. Furthermore, the authors show how stacking 2 or 3 such layers is sufficient to achieve extremely powerful and useful filters.

3.4 Strong Points

- The authors employ a sound, mathematical approach, with each step of their reasoning being clearly explained and motivated, ensuring that the readers understand why their final approach is a sound approximation of the original spectral graph convolutions.
- The very good performance and efficiency of this algorithm made it one of the most well-known approaches within the GRL community, and it often figures as a baseline in more recent works.

3.5 Weak Points

- The presence of so many approximations and reductions in complexity leads some people to argue that using this approach is very different from using spectral graph convolutions, and that it might fail to capture certain aspects. Indeed, this approach can be considered a particular instance of Spatial-Based GCNNs, which are a family of GCNNs that are not based on the mathematical foundation of spectral graph theory.

4 Inductive Representation Learning on Large Graphs

4.1 Problem Statement

Most GNNs are trained by propagating the intermediate embeddings across the whole neighbourhood of each node (which might be defined differently from the actual neighbourhood in the original graph). This has two implications. First of all, it does not scale to large small-world or scale-free networks, because a few layers of propagation may already amount to a large portion of the graph, due to the low diameter of such graphs. Second, all nodes should be present during training, in order to correctly propagate the information.

4.2 Main Contributions

This paper introduces GraphSAGE, a GNN model that easily scales to very large graphs and that generalizes to nodes that were not present during training.

4.3 Solution

The main differenting aspect of GraphSAGE is the use of neighbourhood sampling. Instead of aggregating the embeddings from all neighbours, which in turn aggregated them from their entire neighbourhoods, only a few neighbours are selected to perform the aggregation, and these few in turn select only a few of

their own. Thus complexity is kept low and the number of nodes to aggregate from is bound to a low number.

Furthermore, the paper proposes and evaluates different techniques to perform the aggregation (such as a simple mean, an LSTM and a pooling layer), and also provides two different training modes (unsupervised and semi-supervised).

4.4 Strong Points

- GraphSAGE is one of the most scalable GRL approaches to date, as its complexity is not affected by neighbourhood size. This has the additional advantage of removing the complexity imbalance caused by high-degree nodes, which in certain types of networks may have a few orders of magnitude more neighbours, and thus require much more computation time in each step. With GraphSAGE, all nodes require the same amount of computation in each step, and thus in a parallel or distributed implementation there is no risk of having most workers idle while a few of them deal with the biggest hubs.
- GraphSAGE has achieved an incredible level of popularity, not only within the GRL community, but also in many other fields; it is often the go-to solution for researchers that need to perform ML tasks on graph-structured data and are not themselves expert in the field. Therefore, GraphSAGE has definitely popularized GRL to the wider research community, to a great benefit.

4.5 Weak Points

- The authors consider GraphSAGE to be the first truly inductive approach, that can generalize to nodes not present during training. However, it could be argued that any GCNN could provide this feature, simply by having each node store its intermediate embeddings, and have the new node run the various layers of aggregation locally, as it only needs to talk to its neighbours. This would not provide the same results as a re-training, because in this way the new node does not have a chance to affect the embeddings of its neighbours. However, the same is true in GraphSAGE: without a full retraining, the new node does not have a chance to be sampled by its neighbours. So it could be argued that GraphSAGE is no more inductive than any other GCNN.
- The sampling strategy employed by GraphSAGE does not have any mathematical foundation, and may introduce bias in the training process. Low-degree nodes can very quickly sample their entire neighbourhoods, while high-degree nodes may never have a chance to aggregate information from certain neighbours. Some neighbours might be more important than others in defining certain features of the target node, and yet they may never be sampled due to the randomness of the process.

References

[1] David K. Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129 – 150, 2011.

[2] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.