Project Report

Graph Attention Networks

Justin Pauckert

Supervised by

Prof. Sebastian Pokutta

Christoph Graczyk

July 3, 2021

Seminar on Discrete Optimization and Machine Learning

Berlin Institute of Technology

Introduction

As machine learning models find their way in more and more fields of science, allowing them to work with graph data has been a research topic with increasing popularity in recent years. Graphs are commonly used across various fields such as social science (social networks), natural science (protein-protein interactions) and knowledge graphs [Zhou et al., 2021]. In order to allow for this unique data structure, special neural networks have been developed, called **graph neural networks (GNNs)**. With advancements in the field of deep learning, especially the invention and success of convolutional neural networks, the development of similar methods for graphs suggests itself. In both cases, extracting localized features and composing them to more complex and expressive representations is a major step towards high-performance models for classification and clustering tasks. One way of generalizing convolutions in the graph domain are so-called spatial methods ([Hamilton et al., 2018], [Monti et al., 2016]), where convolutions are defined directly on the graph, operating on groups of spatially close neighbors. Inspired by this idea, [Veličković et al., 2018] proposed a new architecture to compute node representations by attending over a node's neighbors and assessing their individual importance, called **graph attention networks (GATs)**. This report covers a short explanation of the proposed idea and discusses some of its benefits in specific applications. Finally, we will give highlight related developments in the field, including concerns over prevalent data sets used for benchmarks and limitations of architectures like GATs.

GAT Architecture

Graph attention networks are build by stacking graph attentional layers, which we will briefly explain in this section, closely following [Veličković et al., 2018]. The input of a layer is a set of node features, $h = \{h_1, ..., h_N\}$, with $h_i \in \mathbb{R}^F$ and where N is the number of nodes, F the number of features in each node. The layer produces a new feature representation $h_i' \in \mathbb{R}^{F'}$ for each node h_i, possibly of different cardinality F'. To that end, a learnable weight matrix $\mathbf{W} \in \mathbb{R}^{F' \times F}$ and a shared attention mechanism $a : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \to \mathbb{R}$ are introduced. The importance of a node's neighbor, called *attention coefficient*, is then computed as $a(\mathbf{W}h_i, \mathbf{W}h_j)$. Denoting the normalized coefficients as α_{ij}, the hidden state of a node h_i can then be obtained by

$$h_i' = \sigma(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W} h_j)$$

where $\mathcal{N}_i$ is the set of indices for neighbors of h_i and σ a nonlinear function. This process can be initiated multiple times, resulting in multiple attention mechanisms. This popular method is called *multi-head attention* and was adapted from [Vaswani et al., 2017]. The resulting features can then be concatenated or averaged to improve the models performance as illustrated in Figure 1.

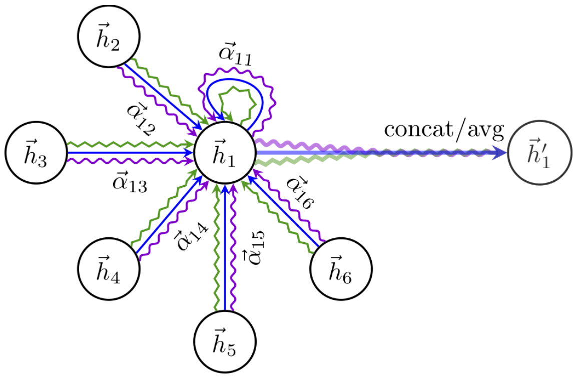

Figure 1: Multi-head attention with three heads by node h_1. [Veličković et al., 2018]

Allthough this architecture has similarities with prior methods, it solves several issues with its flexibility. We will refer to the original paper for more details but mention two desirable properties that will be important for the subsequent chapters.

- Assigning different importances to nodes of the same neighborhood enables a leap in complexity when compared to graph convolutional networks (GCNs). Furthermore, analyzing the learned attention weights may improve **explainability** of the model.
- Since attention mechanism and feature weight matrix are applied in a shared manner, a trained GAT can be applied to **inductive** tasks. That is, they can be evaluated on graphs that were completely unseen during training. Many previous approaches depended on upfront access to the whole graph, making them unfeasible for such tasks.

Applications

To demonstrate the impact of GATs, we will present two examples for their applications. The first one was developed in collaboration with one of the involved authors shortly after the original paper was published. Then, a fairly recent publication will highlight the relevance of the architecture to this day.

GATs for Brain Mesh Segmentation

In [Cucurull et al., 2018], both GCNs and GATs were cutting edge technologies that were assessed against previous mesh parcellation models. Reconstructions of the cortical surface were represented in a graph which had to be divided into areas. Nodes correspond to locations on the brain surface, their features included cortical thickness, curvature and functional connectivity.

The authors considered three kinds of information the models could use: *neighborhood information*, *node features* and *global information*, that is, access to feature information from all the nodes. Previous approaches for this task were unable to exploit all three kinds of information and were therefor limited in their complexity. Alongside GCNs, GATs outperformed all state-of-the-art models, showing the importance of utilizing the mesh structure in the data. Furthermore, the authors showed that dynamic attention mechanisms actually improve performance when compared to a constant attention for all neighbors.

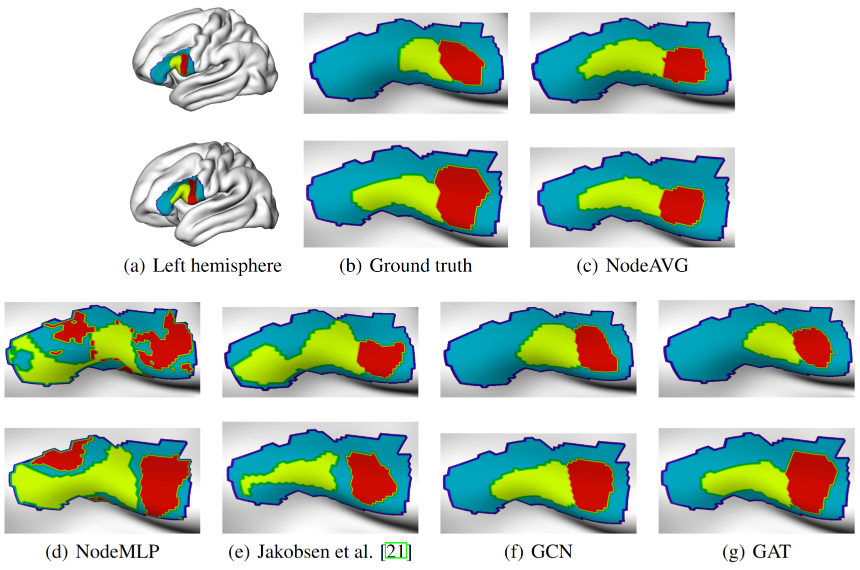

Figure 2: Qualitative area parcellation results for two test set subjects. [Cucurull et al., 2018]

In conclusion, this application demonstrated the potential of GNNs and their applicability to real-world problems. GATs showed promising results, raising hopes for applications in other areas. As we will see later on, this assessment is still up to date and in-line with recent benchmarks.

Improving Explainability through Attention Mechanisms

Visual Question Answering (VQA) is an important task to build systems that better understand the relationship between vision and language by learning to answer questions about an image. The TextVQA dataset (see [Singh et al., 2019]) includes image-question pairs that require models to read text in images and answer questions accordingly.

A recent publication by [Rao et al., 2021] proposes a model that is able to generate explanations for its answers. These are ”consistent with human interpretation, help justify the models' decision and provide useful insights to help diagnose an incorrect prediction”. To that end, they build a graph to encode the relatoinship between objects and text. A GAT is then used to variably weigh an object's adjacent text nodes based on relevancy. The attention weights can be visualized as a heat map, highlighting the position of text tokens that were considered important.

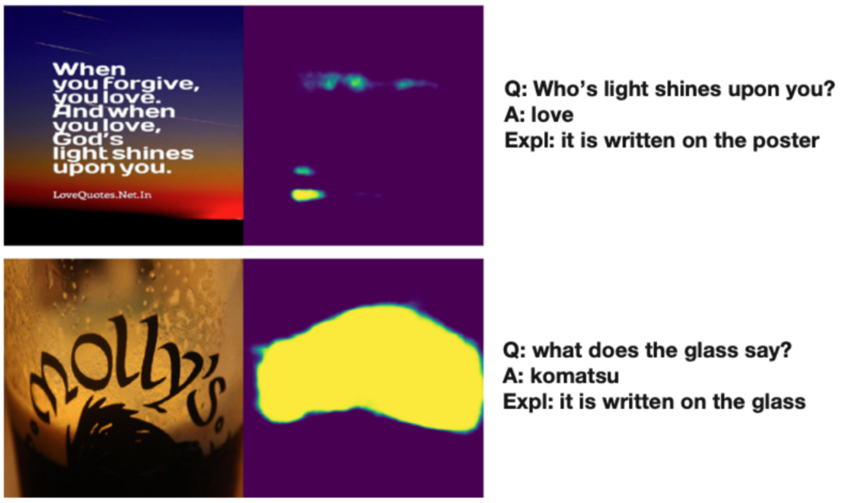
When you forgive, you love. And when you love, God's light shines upon you.
LoveQuotes.Net.In
Q: Who's light shines upon you?
A: love
Expl: it is written on the poster
molly's
Q: what does the glass say?
A: komatsu
Expl: it is written on the glass

Figure 3: Attention weights help explain incorrect decisions. In the first image, the model pays attention to the wrong text tokens. For the second image attention is where it needs to be but the OCR engine fails to read the text correctly. [Rao et al., 2021]

Allthough there are multiple components to this model, it shows that attention weights can help explain its answers. Therefor, this paper is evidence for the claim about increased interpretability through analysis of attention weights made in [Veličković et al., 2018]. Once more, it is shown that GATs are still used today and remain an important tool in the graph neural network domain.

Benchmarking GNNs

With the advancements driven by successful architectures like GATs and GCNs, analyzing and learning from graph data has become a more popular topic. GNNs have become a standard toolkit in the field and new ideas are developed in an increasing pace. In an attempt to increase comparability between architectures, [Dwivedi et al., 2020] proposes a new benchmarking framework and point out flaws in common practices. The question how to fairly evaluate GNN architectures is a complicated one and outside the scope of this report. However, we'll focus on data sets and pass some criticism on evaluation methods used in the orginal GAT paper.

To begin with, data sets used for transductive tasks, namely Cora and Citeseer, are not large enough. According to the authors, small scale data sets "can become a liability in the long run as new GNN models will be designed to overfit the small test sets instead of searching for more generalizable architectures". As a result of this, all GNNs considered in the paper performed almost statistically the same on them. To find differences in performance, one should consider larger datasets. It is worth noting that [Veličković et al., 2018] included the Pubmed dataset as well, which includes about 20k nodes and would still be considered small-scale. All "medium-scale" data sets proposed by the authors contain mutliple graphs with more than a million nodes total, except for a single social network graph with about 236k nodes.

Additionally, models should be evaluated on different tasks. The authors presented 7 datasets to account for the variability of machine learning tasks on graphs. When compared to other models, GAT often performs slightly worse than state-of-the-art competitors, but manages to deliver the best results within the class of GCNs. However, some previously considered inferior architectures like MoNet outmatch GATs when compared on the right dataset. Comparisons for node classification on the PATTERN dataset shows a 10 percent accuracy advantage of MoNet over GAT, see Figure 4.

Still, the authors conclude that GCN models[1] like GAT outperform more recent GNNs which were designed to better distinguish between graph structures. Additionally, models that assign different importance to neighbors in the aggregation step like GAT yielded the best results. Given these promising results, they argue that analyzing the expressivity of attention mechanisms would be a good subject for future research.

[1]With the development of more diverse architectures, message-parsing GNNs based on neighborhood aggregation have been considered a *class* of models, labeled GCNs. Among others, this includes GAT, MoNet and the vanilla GCN proposed by [Kipf and Welling, 2017].

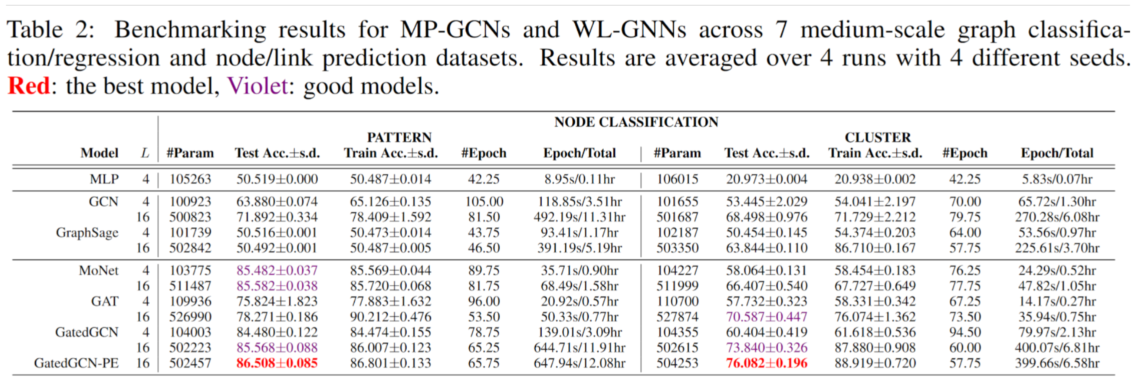

Table 2: Benchmarking results for MP-GCNs and WL-GNNs across 7 medium-scale graph classification/regression and node/link prediction datasets. Results are averaged over 4 runs with 4 different seeds. **Red**: the best model, Violet: good models.

		NODE CLASSIFICATION									
				PATTERN					CLUSTER		
Model	*L*	**#Param**	**Test Acc.±s.d.**	**Train Acc.±s.d.**	**#Epoch**	**Epoch/Total**	**#Param**	**Test Acc.±s.d.**	**Train Acc.±s.d.**	**#Epoch**	**Epoch/Total**
MLP	4	105263	50.519±0.000	50.487±0.014	42.25	8.95s/0.11hr	106015	20.973±0.004	20.938±0.002	42.25	5.83s/0.07hr
GCN	4	100923	63.880±0.074	65.126±0.135	105.00	118.85s/3.51hr	101655	53.445±2.029	54.041±2.197	70.00	65.72s/1.30hr
	16	500823	71.892±0.334	78.409±1.592	81.50	492.19s/11.31hr	501687	68.498±0.976	71.729±2.212	79.75	270.28s/6.08hr
GraphSage	4	101739	50.516±0.001	50.473±0.014	43.75	93.41s/1.17hr	102187	50.454±0.145	54.374±0.203	64.00	53.56s/0.97hr
	16	502842	50.492±0.001	50.487±0.005	46.50	391.19s/5.19hr	503350	63.844±0.110	86.710±0.167	57.75	225.61s/3.70hr
MoNet	4	103775	85.482±0.037	85.569±0.044	89.75	35.71s/0.90hr	104227	58.064±0.131	58.454±0.183	76.25	24.29s/0.52hr
	16	511487	85.582±0.038	85.720±0.068	81.75	68.49s/1.58hr	511999	66.407±0.540	67.727±0.649	77.75	47.82s/1.05hr
GAT	4	109936	75.824±1.823	77.883±1.632	96.00	20.92s/0.57hr	110700	57.732±0.323	58.331±0.342	67.25	14.17s/0.27hr
	16	526990	78.271±0.186	90.212±0.476	53.50	50.33s/0.77hr	527874	70.587±0.447	76.074±1.362	73.50	35.94s/0.75hr
GatedGCN	4	104003	84.480±0.122	84.474±0.155	78.75	139.01s/3.09hr	104355	60.404±0.419	61.618±0.536	94.50	79.97s/2.13hr
	16	502223	85.568±0.088	86.007±0.123	65.25	644.71s/11.91hr	502615	73.840±0.326	87.880±0.908	60.00	400.07s/6.81hr
GatedGCN-PE	16	502457	**86.508±0.085**	86.801±0.133	65.75	647.94s/12.08hr	504253	**76.082±0.196**	88.919±0.720	57.75	399.66s/6.58hr

Figure 4: GAT performs worse than MoNet on specific tasks. [Dwivedi et al., 2020]

Going deep with GNNs

Given the similarities of GNNs with convolutional neural networks, one might be tempted to build deep graph neural networks. Breakthroughs in deep learning such as ResNet ([He et al., 2015]) managed to build models with hundreds of layers, showing that improvement goes hand in hand with depth. Unfortunately, following the same course of action for GNNs does not seem to be beneficial for graph data, allthough they are often labeled as *deep* learning methods. Most models, including the examples discussed in this report, only use a handful of layers and observe worse results with added depth.

Analysis of this issue has provided insight into a few phenomenas specific to the GNN domain. One of them is *over-smoothing*. By adding more and more convolutional layers, a nodes' feature is influenced by a growing number of more distant neighbors. At some point, the receptive field of model is too large and causes all node features to converge to the same output, making it hard to distinguish between them [Oono and Suzuki, 2021].

The effect of information loss with increased depth is even stronger when the neighborhood of a node grows rapidly within a few hops. In this case, there is too much information accumulated into the fixed-size feature vector of a node. This is the so-called *bottleneck* phenomenon.

Both behaviours prevent GATs to improve by simply stacking hundreds of layers. However, there is an important distinction to make with regard to the degree to which this is problematic. In some uses cases, attending short-range neighborhoods already provides all necessary information and more distant information would not contribute to the outcome.

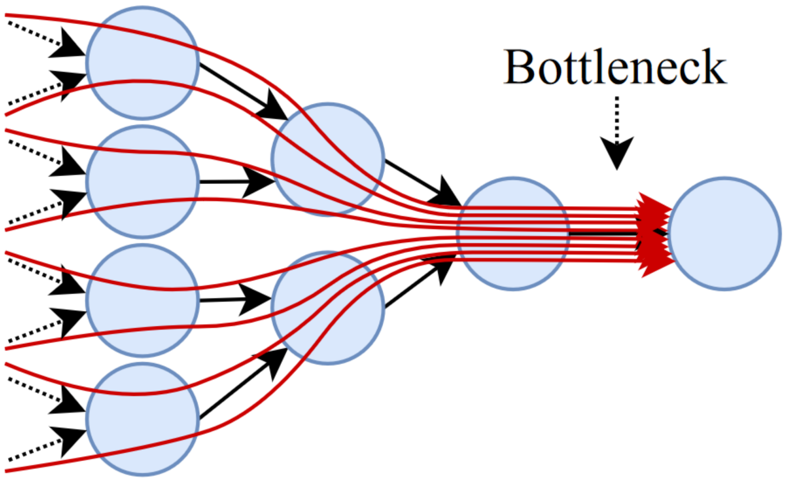

Figure 5: Too much information is aggregated into a feature vector of a single node, resulting in information loss. [Alon and Yahav, 2021]

A typical example are social networks, where link predictions only rely on connections that are a few hops away. Accordingly, GATs are well suited to solve short range taks and would not benefit from more distant information in the first place. However, there are problems which depend on long-range node interactions. Chemical properties of a molecule can depend on atoms on opposite sites of it, making it important to propagate this information by adding more layers [Alon and Yahav, 2021].

Conclusion

Graph Attention Networks were among the first architectures to combine the structural nature of graph data with node-level information. They have proven to be applicable to real-world problems and still hold up to todays evaluation standards. As promising as they are, recent publications indicate that Graph Convolutional Networks in general are limited and significant improvements or entirely new architectures are needed to find solutions to long-range problems.

References

[Alon and Yahav, 2021] Alon, U. and Yahav, E. (2021). On the bottleneck of graph neural networks and its practical implications.

[Cucurull et al., 2018] Cucurull, G., Wagstyl, K., Casanova, A., Velickovic, P., Jakobsen, E., Drozdzal, M., Romero, A., Evans, A. C., and Bengio, Y. (2018). Convolutional neural networks for mesh-based parcellation of the cerebral cortex.

[Dwivedi et al., 2020] Dwivedi, V. P., Joshi, C. K., Laurent, T., Bengio, Y., and Bresson, X. (2020). Benchmarking graph neural networks.

[Hamilton et al., 2018] Hamilton, W. L., Ying, R., and Leskovec, J. (2018). Inductive representation learning on large graphs.

[He et al., 2015] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition.

[Kipf and Welling, 2017] Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks.

[Monti et al., 2016] Monti, F., Boscaini, D., Masci, J., Rodolà, E., Svoboda, J., and Bronstein, M. M. (2016). Geometric deep learning on graphs and manifolds using mixture model cnns.

[Oono and Suzuki, 2021] Oono, K. and Suzuki, T. (2021). Graph neural networks exponentially lose expressive power for node classification.

[Rao et al., 2021] Rao, V. N., Zhen, X., Hovsepian, K., and Shen, M. (2021). A first look: Towards explainable textvqa models via visual and textual explanations.

[Singh et al., 2019] Singh, A., Natarajan, V., Shah, M., Jiang, Y., Chen, X., Batra, D., Parikh, D., and Rohrbach, M. (2019). Towards vqa models that can read.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need.

[Veličković et al., 2018] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph attention networks.

[Zhou et al., 2021] Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. (2021). Graph neural networks: A review of methods and applications.