Project Report

Graph Attention Networks

Justin Pauckert

Supervised by

Prof. Sebastian Pokutta

Christoph Graczyk

June 25, 2021

Seminar on Discrete Optimization and Machine Learning

Berlin Institute of Technology

Introduction

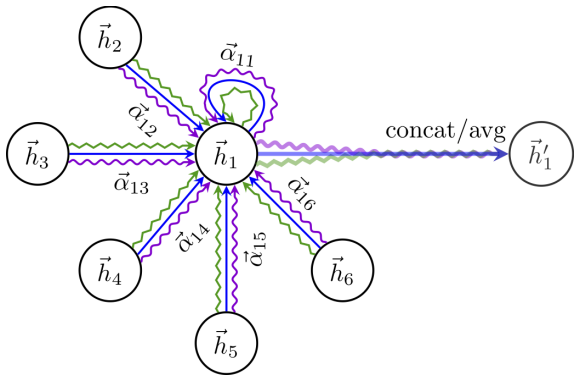
As machine learning models find their way in more and more fields of science, allowing them to work with graph data has been a research topic with increasing popularity in recent years. Graphs are commonly used across various fields such as social science (social networks), natural science (protein-protein 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, $h = \{h_1, ..., h_N\}, h_i \in \mathbb{R}^F$, where N is the number of nodes and 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.



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

Allthough this architexture 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.

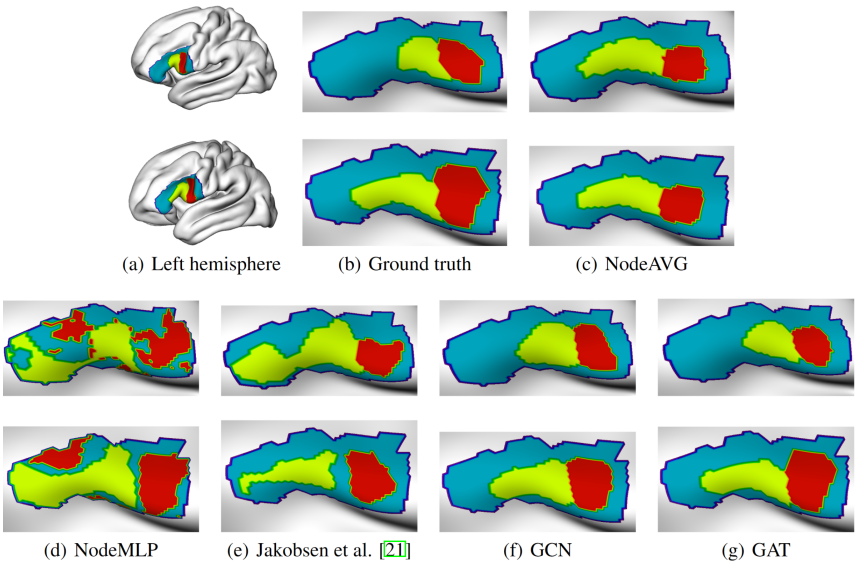
Applications

To demonstrate the impact of GATs, we will present two examples for their applications. The first one was developed in collaboration with some 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 that 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.



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.

Benchmarking GNNs

benchmarks in Question depth of GNNs the road ahead?
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, $h = \{h_1, ..., h_N\}, h_i \in \mathbb{R}^F$, where N is the number of nodes and 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

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

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, $h = \{h_1, ..., h_N\}, h_i \in \mathbb{R}^F$, where N is the number of nodes and 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

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

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, $h = \{h_1, ..., h_N\}, h_i \in \mathbb{R}^F$, where N is the number of nodes and 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

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

Going deep with GNNs

benchmarks in Question depth of GNNs the road ahead?

References

[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.

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

[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.

[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.