Reproduction of a Paper: "How Attentive Are Graph Attention Networks?"

Vladimir Radenković,
Belgrade, Serbia, 11000, vladimirradenkovic27@gmail.com

1 Introduction

Graph Attention Networks (GATs) [2] are a widely used Graph Neural Netowrk (GNN) architectures that has achieved state-of-the-art performance in graph representation learning. The paper "How Attentive Are Graph Attention Networks" [1] argues that GAT employ *static* attention mechanism which limits the ability of GATs to express more complex relationships in graphs. To address this limitation, the paper proposes GATv2, a *dynamic* graph attention variant that modifies the order of operations to enable a strictly more expressive attention mechanism.

The primary objective of this project is to implement both GAT and GATv2 models, compare dynamic and static attention mechanisms and replicate the experiments used in the paper for the evaluating their performance. This includes providing descriptions of the model implementations, datasets, and experiment settings and objectives and reproducing the experimental results in a consistent environment.

2 Model Overview and Theoretical Background

Attention is a mechanism for computing a distribution over input *key* vectors, given a *query* vector, representing the keys' significance or ranking relative to the query. *Static* attention functions always weighs (prioritizes) one key over others, unconditioned on the query. In contrast, dynamic attention can select any key for any query. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with node representations $\mathbf{h_1}, ..., \mathbf{h_n}$ and **a** and **W** the GAT's parameters. GAT's scoring function is:

$$e(\mathbf{h_i}, \mathbf{h_j}) = \text{LeakyReLU}(\mathbf{a_1}^\top \mathbf{W} \mathbf{h_i} + \mathbf{a_2}^\top \mathbf{W} \mathbf{h_j}) \tag{1}$$

where $\mathbf{a} = [\mathbf{a_1} \parallel \mathbf{a_2}] \in \mathbb{R}^{2d'}$ with $\mathbf{a_1}, \mathbf{a_2} \in \mathbb{R}^{d'}$. A node $j_{max} \in \mathcal{V}$ maximizes $\mathbf{a_2}^\top \mathbf{W} \mathbf{h_{j_{max}}}$ among all nodes $j \in \mathcal{V}$. Consequently key $\mathbf{h_{j_{max}}}$ is selected for every query $\mathbf{h_i}$. For any set of nodes $\mathcal{V}$ and a trained GAT layer, the attention function defines a constant ranking of the nodes independent of the query nodes *i*. This is why GAT's attention function computes *static* attention. GATv2 resolves this limitation by applying **a** after the nonlinearity (LeakyReLU) and **W** after the concatenation:

$$e(\mathbf{h_i}, \mathbf{h_j}) = \mathbf{a}^\top \text{LeakyReLU}(\mathbf{W} \cdot [\mathbf{h_i} \parallel \mathbf{h_j}]) \tag{2}$$

By altering the sequence of internal operations in GAT, the scoring function of GATv2 enables the layer to compute attention distributions with varying rankings of the neighbouring nodes and allows the queries to focus on different relevant inputs, making GATv2 strictly more powerful than GAT.

3 Evaluation and experimental Settings

The goal of the experiments was to evaluate dynamic and static attention mechanisms across tasks, using identical model architectures. GAT and GATv2 were compared on node prediction, robustness to noise, link prediction, and the synthetic DictionaryLookup problem.

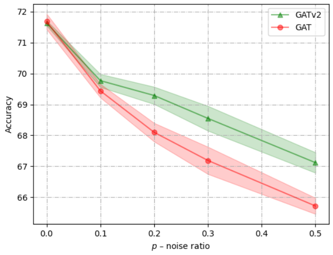


Figure 1: Average test accuracy on ogbn-arxiv compared to the noise ratio after 5 runs ± std.

Model	Attn. Heads	Average Accuracy
GCN	0	71.74 ± 0.29
GraphSAGE	0	71.49 ± 0.27
GAT	1	**71.93** ± 0.27
	8	71.68 ± 0.25
GATv2	1	71.81 ± 0.35
	8	71.63 ± 0.08

Table 1: Average accuracy in ogbn-arxiv dataset after 5 runs ± std.

3.1 Node Prediction

The experiments compared the performance of GAT, GATv2, GCN[5], and GraphSAGE[6] on the ogbn-arxiv dataset. Performance was measured by average accuracy over 5 runs. The results showed no clear

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

superiority of GATv2 over GAT in the node prediction task, with GAT (1 head) outperforming all other models. All models achieved similar accuracy.The dataset might not have complex enough patterns to demonstrate GATv2's advantages. Further research is needed to understand when GATv2's increased expressiveness provides significant performance benefits over GAT for node prediction tasks.

3.2 Robustness to Noise

The robustness to noise was tested on the ogbn-arxiv dataset with induced structural noise. Figure 1 shows that as the noise ratio increased, both GAT and GATv2 showed a decrease in test accuracy. However, GATv2's decline was more gradual due to its dynamic attention ability, helping differentiate between original and noise edges. These results highlight the robustness of dynamic attention over static attention in noisy settings.

3.3 Link Prediction

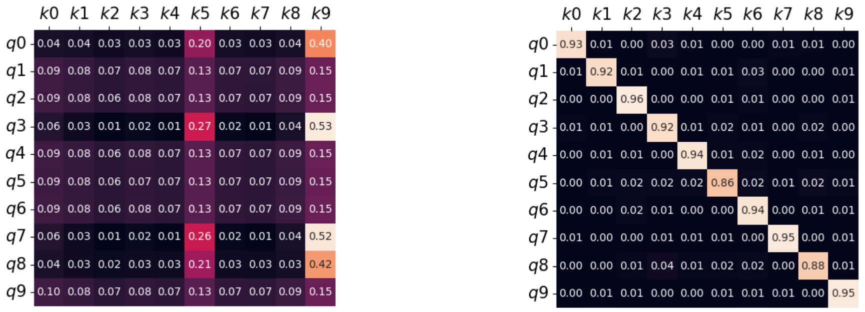
The models were compared on the ogbl-collab dataset using average Hits@10, Hits@50, and Hits@100 metrics after 10 runs, with and without validation edges. GATv2, GAT, and two non-attentive GNNs were evaluated. The non-attentive GraphSAGE outperformed all attentive GNNs. Many GNNs, particularly MPNNs, perform poorly on link prediction tasks due to the equivalence of message passing to the Weisfeiler-Leman graph isomorphism test [8; 9]. GATv2 surpassed GAT for Hits@50 and Hits@100, while their performance for Hits@10 was similar, indicating the limitations of MPNNs for this task.

3.4 Synthetic Benchmark: DictionaryLookup

The DictionaryLookup problem is a synthetic benchmark designed to demonstrate the limitations of static attention mechanisms. It consists of a complete bipartite graph with 2k nodes, including key nodes with attributes and values and query nodes with attributes only. The goal is to predict the value of each query node based on its attribute. This problem is relevant to any subgraph with keys that share more than one query, as each query needs to attend to the keys differently. Experiments were conducted with GATv2 and GAT models, using 1 and 8 attention heads and k values of 10, 20, 30, 40, 50. GAT models with 1 and 8 attention heads failed to fit the training set for any k value, demonstrating their inability to generalize. In contrast, GATv2 achieved 100% train and test accuracy on the node prediction task. For graph prediction tasks, GATv2 with a single attention head reached 100% accuracy on both training and testing sets, while the version with 8 heads achieved near-perfect accuracy. These findings highlight the limitations of the original GAT model and the improvements offered by GATv2 due to its dynamic attention capability.

4 Conclusion

This reproduction project confirmed the original paper's findings, demonstrating that GATv2 outperforms GAT in most benchmarks. GATv2's dynamic attention mechanism proved more robust to noise, while also achieving 100% accuracy on the synthetic DictionaryLookup problem. The project emphasizes the importance of selecting the appropriate graph attention mechanism for a specific problem, as more expressive mechanisms like GATv2 can lead to significant performance improvements. The results successfully validated the original paper's findings. By providing clear analysis of the paper and documentation and code explanations, this work offers valuable insights and enables future researchers to build upon it more efficiently. My code and full project report is publicly available on GitHub.[1]



(a) attention in standard GAT

(b) Attention in GATv2

Figure 2: DictLookup:In a complete bipartite graph of "querry nodes: $\{q0, ..., q9\}$" and "key nodes: $\{k0, ..., k9\}$ standard GAT (Figure 1a) computes *static* attention - the ranking of attention coefficients is global for all nodes in the graph, and is unconditioned on the query node. In contrast, GATv2 (Figure 1b) can actually compute dynamic attention, where every query has a different ranking of attention coefficients of the keys.

[1] https://github.com/VladimirRadenkovic/Reproduction-How-Attentive-Are-Graph-Attention-Networks-

[1] Shaked Brody, Uri Alon, Eran Yahav. How attentive are Graph Attention Networks? In *ICLR*, 2022.

[2] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *Advances in Neural Information Processing Systems*,33,2020.

[3] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020.

[4] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method. *arXiv preprint arXiv:1907.04931*, 2019.

[5] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[6] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in neural information processing systems*, pages 1024–1034, 2017.

[7] Benjamin P. Chamberlain, Sergey Shirobokov, Emanuele Rossi, Fabrizio Frasca, Thomas Markovich, Nils Hammerla, Michael M. Bronstein, Max Hasmire. Graph Neural Networks for Link Prediction with Subgraph Sketching. *arXiv preprint arXiv:2209.15486*, 2022.

[8] 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 *AAAI Conference on Artificial Intelligence*, pages 4602–4609. AAAI Press, 2019.

[9] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019.

[10] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Velicković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges, 2021

[11] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020. benchmarkGNNsVijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982, 2020*

[12] Aleksa Gordić. pytorch-gat. `https://github.com/gordicaleksa/pytorch-GAT`, 2020

[13] Chaitanya Joshi. Transformers are graph neural networks. *The Gradient*, 2020

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. *Attention is all you need. In Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[15] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán AspuruGuzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*, pages 2224–2232, 2015.