Monotonous Monotonicity: Activation Functions in Graph Attention Mechanisms

Abstract

Monotonic activation functions within graph attention mechanisms preserve rank-ordering of static edge-based attention weights. Invoking definitions of expressivity from Brody et al. [3], we explore the impact of non-monotonic activation functions on static and dynamic graph attention mechanisms, achieving new benchmark results on a graph classification task in the process. We empirically and theoretically show that non-monotonic activation functions can generate dynamic attention weights from underlyingly static mechanisms, while also introducing a novel attention function: DYNAMAT.

1 Introduction

One of the first versions of edge-attention in graph neural networks (GNNs) was introduced by Veličković et al. [15] through the GAT model. This approach to GNNs provided a computationally efficient method for assigning variable weights to distinct nodes within the neighborhood of a query node before applying subsequent aggregation or propagation layers. The expressivity of this attention mechanism was directly improved with GATv2 [3] and combined with hop-based attention models [4], [12] via AERO-GNN [8]. Despite the empirical successes of deep attention-based GNNs and progress towards mitigating over-smoothing concerns [8], little research has been conducted to justify the choice of activation functions implemented within attention scoring functions. In this paper, we investigate the impact of specific nonlinearity choices in attention mechanisms, with a specific focus on enhancing expressivity through non-monotonic nonlinearities.

Original Contributions

The following contributions are original to the best of our knowledge. We provide a contrastive analysis of monotonic and non-monotonic activation functions in foundational graph attention models. During this analysis, we produce new benchmark results on a binary graph classification task. Code in this study uses baseline attentional layers for GAT and GATv2 from PyTorch Geometric [5], with original code written for model variations to both alter attention function design and extract learned weights. Lastly, we introduce a novel GAT variation that generates dynamic attention weights, deemed DYNAMAT.

Related Work

A host of analyses exist on expressivity and generalization capabilities in deep GNNs [6], [16]. While most early work shows empirical benefits of attention models on inductive tasks, recent work has questioned the generalizability of deep attention models [18] [6], citing over-smoothing, over-squashing, and over-correlation. Some existing research covers initialization schema for general attention mechanisms [13] and variance propagation for GNNs [9] separately, but there are significant gaps regarding these techniques for graph attentional layers specifically. Lastly, though much research is dedicated to the development of activation functions [10] [19], even comprehensive surveys of graph neural networks dedicate little time to nonlinearity choices within GNN attentional layers [17].

Sample Conference on Neural Information Processing Systems (NeurIPS 9999), City, Country.

2 Preliminaries

2.1 Notation

Let $\mathcal{G}(N, E)$ denote a graph with N nodes and edge-set E. We denote the set of first-order neighbors of node i as $\mathcal{N}_i$.We use $X_i^{(k)} \in \mathbb{R}^F$ to denote the feature representation of node i at layer k, where each node has an initial representation $X_i^{(0)}$, F is the number of node features, and $X^{(k)} \in \mathbb{R}^{N \times F}$ is the matrix containing all $X_i^{(k)}$. Standard notations for matrix operations are used, with the additional specification that $||$ represents concatenation.

2.2 Attention Mechanism

A single-head edge-attentional layer and subsequent propagation in the message-passing regime can be summarized by the following equations:

$$e(i, j) = a(X_i, X_j) \tag{1}$$

$$\mathcal{A}_{ij} = \phi_a(e_{ij}) \tag{2}$$

$$Z^{(k)} = \phi_z(\mathcal{A}XW) \tag{3}$$

where $a : \mathbb{R}^F \times \mathbb{R}^F \to \mathbb{R}$ is used to obtain the raw attention score $e(i, j)$ for i attending to j, and $\mathcal{A}_{ij}$ denotes the attention weight after applying an activation function. The attention score is only calculated for $j \in \mathcal{N}_i$ and $j = i$; it is otherwise 0. For the conventional choice of $\phi_a =$ softmax, this also acts as a normalizing function and bounds attention scores between 0 and 1.

2.3 Expressivity

Borrowing definitions from Brody et al. [3], the typology used in this paper distinguishes different attention functions by their expressivity. Namely, a *static* attention function learns weights such that the ranking of attention scores is unconditioned on the query node. *Dynamic* attention functions compute attention coefficients that are sensitive to the query node. For attention functions that can be either dynamic or static depending on prerequisite conditions, we introduce the label of *quasi-dynamic*.

The design of attention function a is what distinguishes static, dynamic, and quasi-dynamic attention. For GAT and GATv2, the difference primarily lies in the order of operations:

$$\text{Static attention (GAT)} : \qquad e(i, j) = \phi_e(\mathbf{a}^T \cdot [WX_i||WX_j])$$

$$\text{Dynamic attention (GATv2)} : \qquad e(i, j) = \mathbf{a}^T \cdot \phi_e(W[X_i||X_j])$$

In this case, a is parameterized by the learnable weight matrix W and learnable weight vector $\mathbf{a}$. For quasi-dynamic attention, a more direct formulation of the dot product attention mechanism from Vaswani et al. [14] for graphs serves as a prime example:

$$\text{Quasi-dynamic attention (DPGAT)} : \qquad e(i, j) = (X_i^T Q)(X_j^T K)^T (d_k)^{-1/2}$$

where d_k is a scaling factor based on the dimensionality of the Q and K matrices. In this case, DPGAT is quasi-dynamic because it will produce static attention weights for linearly dependent node representations, while producing dynamic attention weights for the linearly independent case. Proofs for the above expressivity classifications are found in Brody et al. [3].

2.4 Activation Function Monotonicity

There are at least two nonlinearities involved in an attentional layer: ϕ_a (typically used for normalization), and ϕ_z. A third nonlinearity is involved if the design of a includes an additional activation function, ϕ_e. In the original implementation of GAT and GATv2, $\phi_e =$ LeakyReLU and $\phi_a =$ softmax, both of which are monotonic activation functions. The main theorem provided in [3] that proves the limited expressivity of the static attention mechanism relies on the monotonicity of the nonlinearity choices for ϕ_e and ϕ_a. While monotonic activation functions are well-suited for gradient descent, loss landscape simplification, and smoothness/continuity, there is no strict requirement that activation functions be monotonic. In fact, there is empirical evidence justifying the

use of non-monotonic activation functions [7] [10] [19], with the underlying intuition being that the non-monotonicity may mitigate vanishing gradient and over-smoothing problems.

In light of this, we 1) explore the impact of activation function selection on expressivity classification and 2) design a novel dynamic attention mechanism.

3 Analysis

3.1 Expressivity Benefits from Non-monotonic Activation Functions

In the case of GAT, before applying the activation ϕ_e, the linearly weighted attention scores for i attending to j have a rank order unconditional on i [3]. If a monotonic activation function is applied (e.g. LeakyReLU), then this rank ordering is preserved. If a non-monotonic activation is applied, then the rank ordering is not necessarily preserved and dynamic weights can be achieved.

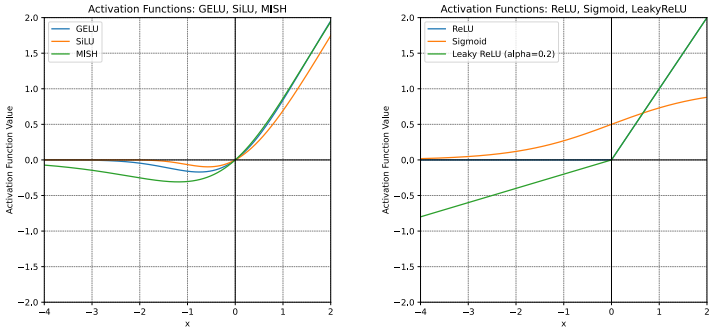

Figure 1: Activation Functions: Non-monotonic (**left**) and monotonic (**right**)

Proof. Consider query node i and two pre-activated attention scores, α_{ij} and α_{ik}, with a non-monotonic activation function ϕ. Select ϕ such that there is exactly one global minimum, with no additional minima or maxima (see GELU, SiLU, or MISH in figure 1 as examples). Define α_c such that $\frac{d\phi}{d\alpha} = 0$ at α_c.

Case 1: $\alpha_{ij} > \alpha_c$ and $\alpha_{ik} > \alpha_c$. For any $\alpha > \alpha_c$, ϕ is monotonically increasing and therefore preserves rank ordering.
Case 2: $\alpha_{ij} < \alpha_c$ and $\alpha_{ik} < \alpha_c$. For any $\alpha < \alpha_c$, ϕ is monotonically decreasing and therefore inverts rank ordering.
Case 3: $\alpha_{ij} < \alpha_c$ and $\alpha_{ik} > \alpha_c$. If two ranked inputs fall on opposite sides of α_c, then the rank ordering will (definitionally) be preserved iff $\phi(\alpha_{ik}) > \phi(\alpha_{ij})$. Stronger claims can be made for more specific choices of activation function. For instance, if $\phi = \text{GELU}$, then rank ordering will be preserved if $\alpha_{ik} > 0$, and may or may not be preserved for $\alpha_c < \alpha_{ik} < 0$.

If conditions for Case 2 or 3 are met, a non-monotonic activation in the attention scoring function thus has the potential to induce dynamic attention weights. □

Following from this proof, conditions for Case 2 or 3 for pre-activated GAT attention scores can be met if it is possible for $\mathbf{a}^T \cdot [WX_i || WX_j] < \alpha_c$. Full proofs governing how best to design the node representations and linear weight initializations to meet these conditions is left as an open question.

3.2 DYNAMAT

We also design and evaluate a new (to the best of our knowledge) dynamic attention mechanism alongside standard versions of GAT and GATv2: **DYNA**mic point-wise **M**ultiplication G**AT** (DYNAMAT).

$$\text{DYNAMAT}: \qquad e(i,j) = \mathbf{a}^T \cdot \phi_e(W(X_i \odot X_j))$$

for $\phi_e =$ GELU and point-wise multiplication $\odot$. Its dynamism can be proven via the same proof for GATv2 found in Brody et al. [3]. Inspired by DPGAT and GATv2, this scoring design seeks to combine the elements of both by directly multiplying corresponding elements of node representations (rather than concatenating), introducing nonlinearity, and finally re-scaling linearly.

4 Experimental Setup

Models were evaluated on graph classification tasks. Base models include vanilla GAT [15] and vanilla GATv2 [3]. Model variations are distinguished by the choice of ϕ_e, with the following variants: GELU, SiLU, MISH, and no activation function (equivalently, $\phi_e(x) = x$). Results are shown in table 1 alongside results for DYNAMAT.

4.1 Datasets

Binary classification datasets include AIDS [11], Mutagenicity [11], and PROTEINS [2]. The ENZYMES [2] dataset is for six-way classification of enzymes. For AIDS and Mutagenicity, nodes represent atoms and edges represent covalent bonds between atoms; each graph represents a molecular compound. Graphs in the AIDS dataset are active/inactive against HIV, while graphs in the Mutagenicity dataset are mutagens/not-mutagens. Nodes from PROTEINS graphs are amino acids, while edges exist for amino acids less than six Angstroms apart; each graph is labeled as enzyme/not-enzyme. ENZYMES are classified by their top level enzyme commission (EC) number.

4.2 Hyperparameters

Hyperparameters remained constant across all models and training runs. Each model was implemented with one attention head for each of three attentional layers. The first two layers employed Exponential Linear Units (ELU) for ϕ_z, while the final layer employed mean pooling with sigmoid activation. Normalization was done by setting $\phi_a =$ softmax. The hidden dimension for all layers was 64. The optimizer was Adam with a learning rate of 0.001 and weight decay of $5 * 10^{-4}$. The loss function utilized was Binary Cross Entropy for binary classification and Cross Entropy for multi-class classification. For binary classification tasks, training consisted of 10 epochs, replicated over 10 training runs. For multi-class classification, training consisted of 100 epochs for 10 training runs.

5 Results

Results for empirical experiments can be seen in table 1. We found that the highest performance for all four tasks was achieved by models with non-monotonic activation functions. In fact, we have established (to the best of our knowledge) a new benchmark result on the binary graph classification task for the AIDS dataset with 98.67% accuracy.

Model	AIDS	Mutagenicity	PROTEINS	ENZYMES
GAT	98.37 ± 0.4	70.32 ± 1.2	64.52 ± 3.7	49.44 ± 3.1
GATv2	98.57 ± 0.5	71.44 ± 0.7	64.46 ± 2.7	51.56 ± 4.9
GAT_GELU	98.50 ± 0.3	70.28 ± 2.2	64.17 ± 2.5	49.67 ± 4.6
GAT_SILU	98.07 ± 0.6	69.74 ± 1.6	65.77 ± 1.5	49.78 ± 2.8
GAT_MISH	98.00 ± 0.7	70.14 ± 1.5	65.36 ± 1.9	49.78 ± 2.6
GAT_NONE	98.33 ± 0.6	70.25 ± 1.6	63.33 ± 2.4	48.89 ± 3.3
GATv2_GELU	98.10 ± 0.5	$\mathbf{71.66 \pm 0.9}$	65.53 ± 3.1	$\mathbf{53.22 \pm 2.9}$
GATv2_SILU	$\mathbf{98.67 \pm 0.3}$	70.87 ± 1.1	64.05 ± 3.2	52.11 ± 3.4
GATv2_MISH	98.57 ± 0.3	71.43 ± 1.0	62.08 ± 2.3	50.67 ± 3.4
GATv2_NONE	98.17 ± 0.8	70.31 ± 2.1	61.13 ± 2.5	49.91 ± 3.0
DYNAMAT	98.37 ± 0.8	70.63 ± 1.0	$\mathbf{68.99 \pm 1.4}$	44.89 ± 7.6
Benchmark	97.3	83	84.91	78.39

Table 1: **Performance on Graph Classification Tasks**. Top results are indicated in yellow, while top results excluding benchmarks are indicated in bold. Results include the mean accuracy and standard deviation across training runs. Benchmarks are recorded to the best of our knowledge [11] [1].

Comparing results within GAT variants, we see that introducing non-monotonicity improved model performance. For 3/4 of the classification tasks, the highest performing GAT variants were those with non-monotonic nonlinearities. To confirm that non-monotonic activation functions can make static graph attention mechanisms dynamic, we extracted the learned attention weights. As can be seen in figure 2, dynamic attention can be and was achieved, as the ranking of weights extracted from GAT_GELU is sensitive to the query node for the sample graph displayed from the AIDS dataset.

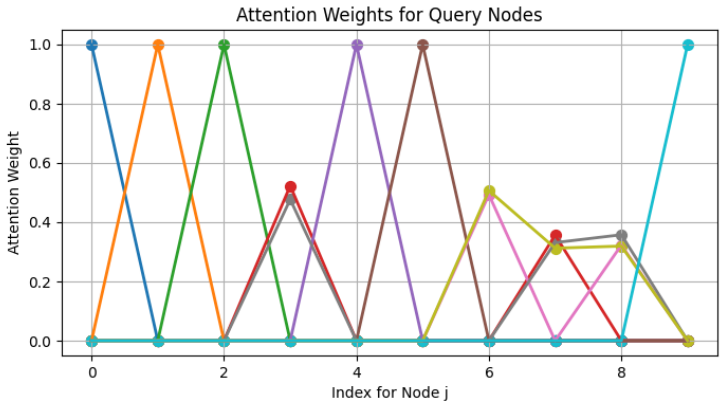

Figure 2: **Dynamically Learned Weights from GAT_GELU**. Colors represent query nodes i from a sample AIDS graph

Interestingly, for GATv2 variants, which already generate dynamic attention coefficients, similar results were observed; for 4/4 classification tasks, the highest performing GATv2 variants were those with non-monotonic nonlinearities. These results imply that even dynamic attention mechanisms may benefit from non-monotonic activation functions. The mathematical formulation as to why this may be the case is left as an open question; however, future studies targeting this question specifically may focus on in-depth analyses of over-smoothing and over-squashing of deep attention mechanisms to see if non-monotonicity mitigates these problems, perhaps invoking over-smoothing vulnerability definitions similar to Lee et al. [8].

Removing ϕ_e entirely had mixed results on model performance across datasets. In the worst case (PROTEINS), GAT_NONE performs 5.66% worse than the top performing model, while in the best case (AIDS), GAT_NONE performs just 0.33% worse than the top performing model. These results may suggest that ϕ_e is not necessary; however, further investigation is needed.

Lastly, DYNAMAT performance was unremarkable for AIDS and Mutagenicity, achieving comparable accuracies as other model variants. For PROTEINS, DYNAMAT was the top performing model, suggesting the structures of the PROTEINS graphs may particularly benefit from point-wise multiplication of node representations. The opposite is true for ENZYMES, as performance is the worst compared to all other variants. These results add to the growing body of literature demonstrating the need for bespoke attention mechanisms depending on the nature of the task and data at hand.

6 Conclusion

In this paper, we analyze the utility of non-monotonic activation functions within graph attention scoring mechanisms, showing the potential for carefully chosen activation functions to generate dynamic weights from previously static mechanisms. We provide empirical evidence on the benefits of non-monotonic activations for both static and dynamic attention, while simultaneously questioning the necessity of additional nonlinearities beyond ϕ_a and ϕ_z. In evaluating our models, we achieved a new benchmark result on the AIDS graph classification task and introduced a new attention scoring function, DYNAMAT, which outperformed all other models on PROTEINS classification. Further work might explore over-smoothing and over-correlation in deep attentional layers, as well as initialization schema that stabilize variance propagation while simultaneously encouraging the learning of dynamic attention weights.

References

[1] Papers with Code - Graph Classification. https://paperswithcode.com/.

[2] Karsten Borgwadt, Cheng Soon Ong, and Hans-Peter Kriegel. Protein function prediction via graph kernels | Bioinformatics | Oxford Academic.

[3] Shaked Brody, Uri Alon, and Eran Yahav. How Attentive are Graph Attention Networks?, January 2022. arXiv:2105.14491 [cs].

[4] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive Universal Generalized PageRank Graph Neural Network, October 2021. arXiv:2006.07988 [cs, stat].

[5] Matthias Fey and Jan Eric Lenssen. Fast Graph Representation Learning with PyTorch Geometric, April 2019. arXiv:1903.02428 [cs, stat].

[6] Boris Knyazev, Graham W Taylor, and Mohamed Amer. Understanding Attention and Generalization in Graph Neural Networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[7] Minhyeok Lee. GELU Activation Function in Deep Learning: A Comprehensive Mathematical Analysis and Performance, August 2023. arXiv:2305.12073 [cs].

[8] Soo Yong Lee, Fanchen Bu, Jaemin Yoo, and Kijung Shin. Towards Deep Attention in Graph Neural Networks: Problems and Remedies. In *Proceedings of the 40th International Conference on Machine Learning*, pages 18774–18795. PMLR, July 2023. ISSN: 2640-3498.

[9] Jiahang Li, Yakun Song, Xiang Song, and David Wipf. On the Initialization of Graph Neural Networks. In *Proceedings of the 40th International Conference on Machine Learning*, pages 19911–19931. PMLR, July 2023. ISSN: 2640-3498.

[10] Diganta Misra. Mish: A Self Regularized Non-Monotonic Activation Function, August 2020. arXiv:1908.08681 [cs, stat].

[11] Kaspar Riesen and Horst Bunke. IAM Graph Database Repository for Graph Based Pattern Recognition and Machine Learning. In Niels da Vitoria Lobo, Takis Kasparis, Fabio Roli, James T. Kwok, Michael Georgiopoulos, Georgios C. Anagnostopoulos, and Marco Loog, editors, *Structural, Syntactic, and Statistical Pattern Recognition*, Lecture Notes in Computer Science, pages 287–297, Berlin, Heidelberg, 2008. Springer.

[12] Veronika Thost and Jie Chen. Directed Acyclic Graph Neural Networks, February 2021. arXiv:2101.07965 [cs].

[13] Asher Trockman and J. Zico Kolter. Mimetic Initialization of Self-Attention Layers. In *Proceedings of the 40th International Conference on Machine Learning*, pages 34456–34468. PMLR, July 2023. ISSN: 2640-3498.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, August 2023. arXiv:1706.03762 [cs].

[15] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks, February 2018. arXiv:1710.10903 [cs, stat].

[16] Xinyi Wu, Amir Ajorlou, Zihui Wu, and Ali Jadbabaie. Demystifying Oversmoothing in Attention-Based Graph Neural Networks, October 2023. arXiv:2305.16102 [cs, stat].

[17] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, January 2021. Conference Name: IEEE Transactions on Neural Networks and Learning Systems.

[18] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How Powerful are Graph Neural Networks?, February 2019. arXiv:1810.00826 [cs, stat].

[19] Hegui Zhu, Huimin Zeng, Jinhai Liu, and Xiangde Zhang. Logish: A new nonlinear nonmonotonic activation function for convolutional neural network. *Neurocomputing*, 458:490–499, October 2021.