Enhancing Attention-based Graph Neural Networks via Cardinality Preservation

Shuo Zhang,[1] Lei Xie[1,2,3]
[1] Ph.D. Program in Computer Science, The Graduate Center, The City University of New York
[2] Department of Computer Science, Hunter College, The City University of New York
[3] Helen and Robert Appel Alzheimer's Disease Research Institute, Feil Family Brain and Mind Research Institute, Weill Cornell Medicine, Cornell University
szhang4@gradcenter.cuny.edu, lei.xie@hunter.cuny.edu

Abstract

GNNs with attention mechanism in aggregation have been actively applied in various fields. Though the attention-based GNNs have achieved remarkable results in various tasks, a clear understanding of their discriminative capacities is missing. In this work, we start from theoretical analysis to understand the discriminative powers of those attention-based GNNs. Our analysis determines all cases when attention-based GNNs can always fail to distinguish certain distinct structures. Those cases appear due to the ignorance of cardinality information in attention-based aggregation. To improve the expression power of attention-based GNNs, we propose cardinality preserved attention (CPA) models for attention-based GNNs. The experiments on node and graph classification confirm our theoretical analysis and show the competitive performance of our CPA models.

Introduction

In recent years, Graph Neural Networks (GNNs) have been proposed to learn the representations of graph-structured data and attract a growing interest (Scarselli et al. 2009; Li et al. 2016; Duvenaud et al. 2015; Niepert, Ahmed, and Kutzkov 2016; Kipf and Welling 2017; Hamilton, Ying, and Leskovec 2017; Zhang et al. 2018; Ying et al. 2018; Morris et al. 2019a; Xu et al. 2019). When iteratively updating node embeddings by aggregating node features and structural information in the graph, most of the GNNs will assign non-parametric weight between nodes (Kipf and Welling 2017; Hamilton, Ying, and Leskovec 2017; Xu et al. 2019). Such aggregators fail to learn the information between a target node and its neighbors, which is important in real-world data since not all edges have similar impacts. To make the edge weights trainable, attention mechanism (Bahdanau, Cho, and Bengio 2014; Vaswani et al. 2017) is incorporated in GNNs and achieves promising performance on various tasks (Veličković et al. 2018; Thekumparampil et al. 2018). However, we are not clear about the exact power of attention mechanism in graph representation learning.

In this work, we make efforts to theoretically analyze the discriminative power of GNNs with attention-based aggregators. Our findings reveal that previous proposed attention-based aggregators fail to distinguish certain distinct structures due to the ignorance of cardinality information in aggregation. Thus we propose models to preserve cardinality in the attention mechanism so as to improve its discriminative power. Our experiments on node/graph classification confirm our theoretical analysis and the improvements from our models. Moreover, our models achieve competitive results on graph classification benchmarks.

Preliminaries

Notations

Let $G = (V, E)$ be a graph with set of nodes V and set of edges E. We denote the set of node i and its nearest neighbors $\mathcal{N}(i)$ as $\tilde{\mathcal{N}}(i) = \mathcal{N}(i) \cup \{i\}$. For the nodes in $\tilde{\mathcal{N}}(i)$, their feature vectors form a *multiset* $M(i) = (S_i, \mu_i)$, where $S_i = \{s_1, \ldots, s_n\}$ is the *ground set* of $M(i)$, and $\mu_i : S_i \to \mathbb{N}^*$ is the *multiplicity function* that gives the *multiplicity* of each $s \in S_i$. The *cardinality* $|M|$ of a multiset is the number of elements (with multiplicity) in the multiset.

Attention-Based GNNs

We focus on the Graph Neural Networks (GNNs) under the massage-passing framework (Wu et al. 2019; Zhou et al. 2018). For attention-based GNNs, attention mechanism is used in the nearest neighbor aggregation (Lee et al. 2018). We can formulate the aggregation in l-th layer as follows:

$$e_{ij}^{l-1} = Att\big(h_i^{l-1}, h_j^{l-1}\big), \tag{1}$$

$$\alpha_{ij}^{l-1} = \mathrm{softmax}\left(e_{ij}^{l-1}\right) = \frac{\exp(e_{ij}^{l-1})}{\sum_{k\in\tilde{\mathcal{N}}(i)} \exp\left(e_{ik}^{l-1}\right)}, \tag{2}$$

$$h_i^l = f^l\Big(\sum\nolimits_{j\in\tilde{\mathcal{N}}(i)} \alpha_{ij}^{l-1} h_j^{l-1}\Big), \tag{3}$$

where attention function Att computes e_{ij}, which is the attention coefficient. α_{ij} is the attention weight and f is a nonlinear function. In the final layer, since the node representation h_i^L after L iterations contains the L-step neighborhood information, it can be directly used for local/node-level

tasks. While for global/graph-level tasks, an extra readout function g can be used to compute the whole graph representation h_G from all h_i^L: $h_G = g\big(\{h_i^L, \forall i \in G\}\big)$.

Related Works

Since GNNs have achieved remarkable results in practice, a clear understanding of the power of GNNs in graph representational learning is needed to design better models and make further improvements. Recent works (Morris et al. 2019b; Xu et al. 2019; Maron et al. 2019) focus on understanding the discriminative power of GNNs by comparing it to the Weisfeiler-Lehman (WL) test (Weisfeiler and Leman 1968) when deciding the graph isomorphism. It is proved that massage-passing-based GNNs which aggregate the nearest neighbor node features of a node for embedding are at most as powerful as the 1-WL test (Xu et al. 2019). Inspired by the higher discriminative power of the k-WL test ($k > 2$) (Cai, Fürer, and Immerman 1992) than the 1-WL test, GNNs that have a theoretically higher discriminative power than the massage-passing-based GNNs have been proposed based on the k-WL test (Morris et al. 2019b; Maron et al. 2019). However, the GNNs proposed in those works don't specifically contain the attention mechanism as the part of their analysis. So it's currently unknown whether the attention mechanism will constrain the discriminative power. Our work focuses on the massage-passing-based GNNs with attention mechanism, which are upper bounded by the 1-WL test.

Another recent work (Knyazev, Taylor, and Amer 2019) aims to understand the attention mechanism over nodes in GNNs with experiments in a controlled environment. However, the attention mechanism discussed in the work is used in the pooling layer for the pooling of nodes, while our work investigates the usage of attention mechanism in the aggregation layer for the updating of nodes.

Limitation of Attention-Based GNNs

In this section, we theoretically analyze the discriminative power of attention-based GNNs and show their limitations. The discriminative power means how well an attention-based GNN can distinguish different elements (local or global structures). We find that previously proposed attention-based GNNs can fail in certain cases and the discriminative power is limited. Besides, by theoretically finding out all cases that always fail an attention-based GNN, we reveal that those failures come from the lack of cardinality preservation in attention-based aggregators. The details of proofs are included in the Supplemental Material.

Discriminative Power of Attention-based GNNs

We assume the node input feature space is countable. For any attention-based GNNs, we give the conditions in Lemma 1 to make them reach the upper bound of discriminative power when distinguishing different elements (local or global structures). In particular, each local structure belongs to a node and is the k-height subtree structure rooted at the node, which is naturally captured in the node feature h_i^k after k iterations in a GNN. The global structure contains the information of all such subtrees in a graph.

Lemma 1. *Let $\mathcal{A} : \mathcal{G} \to \mathbb{R}^g$ be a GNN following the neighborhood aggregation scheme with the attention-based aggregator (Equation 3). For global-level task, an extra readout function is used in the final layer. $\mathcal{A}$ can reach its upper bound of discriminative power (can distinguish all distinct local structures or be as powerful as the 1-WL test when distinguishing distinct global structures) after sufficient iterations with the following conditions:*

- ***Local-level**: Function f and the weighted summation in Equation 3 are injective.*
- ***Global-level**: Besides the conditions for local-level, $\mathcal{A}$'s readout function is injective.*

With Lemma 1, we are interested in whether its conditions can always be satisfied, so as to reach the upper bound of discriminative capacity of an attention-based GNN. Since the function f and the global-level readout function can be predetermined to be injective, we focus on whether the weighted summation function in attention-based aggregator can be injective.

The Non-Injectivity of Attention-Based Aggregator

In this part, we aim to answer the following two questions:

Q 1. *Can the attention-based GNNs actually reach the upper bound of discriminative power? In other words, can the weighted summation function in an attention-based aggregator be injective?*

Q 2. *If not, can we determine **all** of the cases that prevent any kind of weighted summation function being injective?*

In Theorem 1, we answer Q1 with *No* by giving the cases that make the weighted summation function W not to be injective. So that the attention-based GNNs can **never** meet their upper bound of discriminative power, which is stated in Corollary 1. Moreover, we answer Q2 with *Yes* in Theorem 1 by pointing out those cases are the **only** reason to always prevent W being injective. This alleviates the difficulty of summarizing the properties of those cases. Besides, we can specifically propose methods to avoid those cases so as to let W to be injective.

Theorem 1. *Assume the input feature space $\mathcal{X}$ is countable. Given a multiset $X \subset \mathcal{X}$ and the node feature c of the central node, the weighted summation function $h(c, X)$ in aggregation is defined as $h(c, X) = \sum_{x \in X} \alpha_{cx} f(x)$, where $f : \mathcal{X} \to \mathbb{R}^n$ is a mapping of input feature vector and α_{cx} is the attention weight between $f(c)$ and $f(x)$ calculated by the attention function Att in Equation 1 and the softmax function in Equation 2. For all f and Att, $h(c_1, X_1) = h(c_2, X_2)$ **if and only if** $c_1 = c_2$, $X_1 = (S, \mu)$ and $X_2 = (S, k \cdot \mu)$ for $k \in \mathbb{N}^*$. In other words, h will map different multisets to the same embedding if and only if the multisets have the same central node feature and the same distribution of node features.*

Corollary 1. *Let $\mathcal{A}$ be the GNN defined in Lemma 1. $\mathcal{A}$ never reaches its upper bound of discriminative power:*

There exists two different subtrees S_1 and S_2 or two graphs G_1 and G_2 that the Weisfeiler-Lehman test decides as non-isomorphic, such that $\mathcal{A}$ always maps the two subtrees/graphs to the same embeddings.

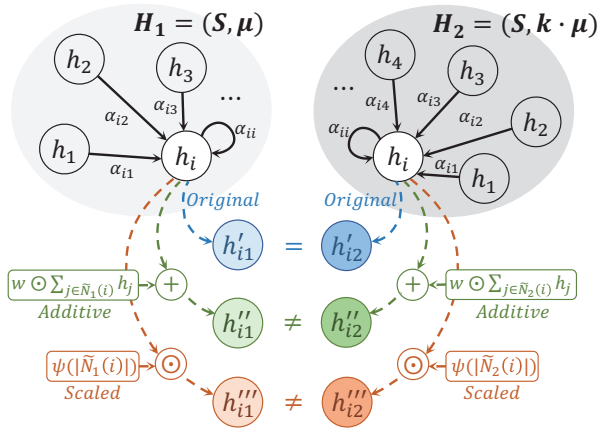

Figure 1: An illustration of different attention-based aggregators on multiset of node features. Given two distinct multisets H_1 and H_2 that have the same central node feature h_i and the same distribution of node features, aggregators will map h_i to h_{i1} and h_{i2} for H_1 and H_2. The *Original* model will get $h'_{i1} = h'_{i2}$ and fail to distinguish H_1 and H_2, while our *Additive* and *Scaled* models can always distinguish H_1 and H_2 with $h''_{i1} \neq h''_{i2}$ and $h'''_{i1} \neq h'''_{i2}$.

Attention Mechanism Fails to Preserve Cardinality

With Theorem 1, we are now interested in the properties of all cases that always prevent the weighted summation functions W being injective. Since the multisets that all W fail to distinguish share the same distribution of node features, we can say that W ignores the multiplicity information of each identical element in the multisets. Thus the cardinality of the multiset is not preserved:

Corollary 2. *Let $\mathcal{A}$ be the GNN defined in Lemma 1. The attention-based aggregator in $\mathcal{A}$ cannot preserve the cardinality information of the multiset of node features in aggregation.*

Cardinality Preserved Attention (CPA) Model

Since the cardinality of the multiset is not preserved in attention-based aggregators, our goal is to propose modifications to any kind of attention mechanism to make them capture the cardinality information. So that all of the cases that always prevent attention-based aggregator being injective can be avoided. To achieve our goal, we modify the *Original* weighted summation function in Equation 3 while keeping the original expressive power in Equation 1:

Model 1. (Additive)

$$h_i^l = f^l\Big(\sum\nolimits_{j\in\tilde{\mathcal{N}}(i)} \alpha_{ij}^{l-1} h_j^{l-1} + w^l \odot \sum\nolimits_{j\in\tilde{\mathcal{N}}(i)} h_j^{l-1}\Big), \tag{4}$$

Model 2. (Scaled)

$$h_i^l = f^l\Big(\psi^l\big(|\tilde{\mathcal{N}}(i)|\big) \odot \sum\nolimits_{j\in\tilde{\mathcal{N}}(i)} \alpha_{ij}^{l-1} h_j^{l-1}\Big), \tag{5}$$

where w is a non-zero vector $\in \mathbb{R}^n$, $\odot$ denotes the element-wise multiplication, $|\tilde{\mathcal{N}}(i)|$ equals to the cardinality of the multiset $\tilde{\mathcal{N}}(i)$, $\psi : \mathbb{Z}^+ \to \mathbb{R}^n$ is an injective function.

In the *Additive* model, each element in the multiset will contribute to the term that we added to preserve the cardinality information. In the *Scaled* model, the original weighted summation is directly multiplied by a representational vector of the cardinality value. We summarize the effect of our models in Corollary 3 and illustrate it in Figure 1.

Corollary 3. *Let $\mathcal{T}$ be the original attention-based aggregator in Equation 3. With our proposed **Cardinality Preserved Attention (CPA)** models in Equation 4 and 5, $\mathcal{T}$'s discriminative power is increased: $\mathcal{T}$ can now distinguish **all** different multisets in aggregation that it previously always fails to distinguish.*

Moreover, we simplify our models by fixing the values in w and $\psi(|\tilde{\mathcal{N}}(i)|)$ and define two CPA variants:

Model 3. (f-Additive)

$$h_i^l = f^l\Big(\sum\nolimits_{j\in\tilde{\mathcal{N}}(i)} (\alpha_{ij}^{l-1} + 1) h_j^{l-1}\Big), \tag{6}$$

Model 4. (f-Scaled)

$$h_i^l = f^l\Big(|\tilde{\mathcal{N}}(i)| \cdot \sum\nolimits_{j\in\tilde{\mathcal{N}}(i)} \alpha_{ij}^{l-1} h_j^{l-1}\Big). \tag{7}$$

Model 3 and 4 still preserve the cardinality information and have the **same** time and space complexity as the *Original* model in Equation 3.

Experiments

In our experiments, we focus on the following questions:

Q 3. *Since attention-based GNNs (e.g. GAT) are originally proposed for local-level tasks like node classification, will those models fail or not meet the upper bound of discriminative power when solving certain node classification tasks? If so, can our proposed CPA models improve the original model?*

Q 4. *For global-level tasks like graph classification, how well can the original attention-based GNNs perform? Can our proposed CPA models improve the original model?*

Q 5. *How the attention-based GNNs with our CPA models perform compared to baselines?*

To answer Question 3, we design a node classification task which is to predict whether or not a node is included in a triangle as one vertex in a graph. To answer Question 4 and 5, we perform experiments on graph classification benchmarks and evaluate the performance of attention-based GNNs with CPA models.

Experimental Setup

Datasets In our synthetic task (TRIANGLE-NODE) for predicting whether or not a node is included in a triangle, we generate a graph with different node features. In our experiment on graph classification, we use 6 benchmark datasets: 2 social network datasets (REDDIT-BINARY (RE-B), REDDIT-MULTI5K (RE-M5K)) and 4 bioinformatics datasets (MUTAG, PROTEINS, ENZYMES, NCI1). More details of the datasets are provided in Supplemental Material.

Table 1: Testing accuracies(%) of GAT variants (the original GAT and the GAT applied with each of our 4 CPA models) on TRIANGLE-NODE dataset for node classification. We highlight the result of the best performed model. The proportion P of multisets that hold the properties in Theorem 1 among all multisets is also reported.

Dataset $P(\%)$	TRIANGLE-NODE 29.2
Original	78.40 ± 7.65
Additive	91.31 ± 1.19
Scaled	$\mathbf{91.38 \pm 1.23}$
f-Additive	91.18 ± 1.24
f-Scaled	91.36 ± 1.26

Table 2: Testing accuracies(%) of GAT-GC variants (the original one and the ones applied with each of our 4 CPA models) on social network datasets. We highlight the result of the best performed model per dataset. The proportion P of multisets that hold the properties in Theorem 1 among all multisets is also reported for each dataset.

Datasets $P(\%)$	RE-B 100.0	RE-M5K 100.0
Original	50.00 ± 0.00	20.00 ± 0.00
Additive	$\mathbf{93.07 \pm 1.82}$	$\mathbf{57.39 \pm 2.09}$
Scaled	92.36 ± 2.27	56.76 ± 2.26
f-Additive	93.05 ± 1.87	56.43 ± 2.38
f-Scaled	92.57 ± 2.06	57.22 ± 2.20

Models In our experiments, the ***Original*** model is the one that uses the original version of an attention mechanism. We apply each of our 4 CPA models (***Additive***, ***Scaled***, ***f-Additive*** and ***f-Scaled***) to the original attention mechanism for comparison. In the *Additive* and *Scaled* models, we take advantage of the approximation capability of multi-layer perceptron (MLP) (Hornik, Stinchcombe, and White 1989; Hornik 1991) to model f and ψ.

For node classification, we use GAT (Veličković et al. 2018) as the *Original* model. For graph classification, we build a GNN (GAT-GC) based on GAT as the *Original* model: We adopt the attention mechanism in GAT to specify the form of Equation 1: $e_{ij} = \mathrm{LeakyReLU}\left(\mathbf{a}^\top \left[\mathbf{W}h_i \| \mathbf{W}h_j\right]\right)$. For the readout function, a naive way is to only consider the node embeddings from the last iteration. Although a sufficient number of iterations can help to avoid the cases in Theorem 1 by aggregating more diverse node features, the features from the latter iterations may generalize worse and the GNNs usually have shallow structures (Xu et al. 2019; Zhou et al. 2018). So the GAT-GC adopts the same function as used in (Xu et al. 2018; 2019; Lee, Lee, and Kang 2019; Li et al. 2019), which concatenates graph embeddings from all iterations: $h_G = \|_{k=0}^{L} \left(\mathrm{Readout}(\{h_i^l | i \in G\})\right)$, Readout function can be sum or mean. With CPA models, the cases in Theorem 1 can be avoided in each iteration. Full experimental settings are included in Supplemental Material.

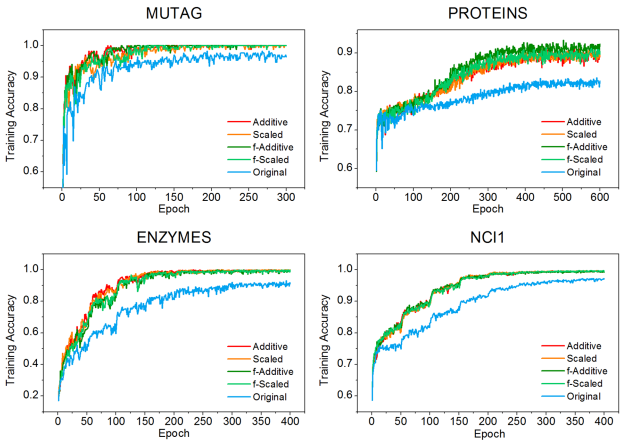

Figure 2: Training curves of GAT-GC variants on bioinformatics datasets.

Table 3: Testing accuracies(%) of GAT-GC variants (the original one and the ones applied with each of our 4 CPA models) on bioinformatics datasets. We highlight the result of the best performed model per dataset. The highlighted results are significantly higher than those from the corresponding *Original* model under paired t-test at significance level 5%. The proportion P of multisets that hold the properties in Theorem 1 among all multisets is also reported for each dataset.

Datasets $P(\%)$	MUTAG 56.9	PROTEINS 29.3	ENZYMES 29.4	NCI1 43.3
Original	84.96 ± 7.65	75.64 ± 3.96	58.08 ± 6.82	80.29 ± 1.89
Additive	89.75 ± 6.39	76.61 ± 3.80	58.90 ± 6.96	81.92 ± 1.89
Scaled	89.65 ± 7.47	76.44 ± 3.77	58.35 ± 6.97	82.18 ± 1.67
f-Additive	90.34 ± 6.05	76.60 ± 3.91	$\mathbf{59.80 \pm 6.18}$	81.96 ± 2.01
f-Scaled	$\mathbf{90.44 \pm 6.44}$	$\mathbf{76.81 \pm 3.77}$	58.45 ± 6.35	$\mathbf{82.28 \pm 1.81}$

Node Classification

For the TRIANGLE-NODE dataset, the proportion P of multisets that hold the properties in Theorem 1 is 29.2%, as shown in Table 1. The classification accuracy of the *Original* model (GAT) is significantly lower than the CPA models. It supports the claim in Corollary 1: the *Original* model fails to distinguish all distinct multisets in the dataset and exhibits constrained discriminate power. On the contrary, CPA models can distinguish all different multisets in the graph as suggested in Corollary 3 and indeed significantly improve the accuracy of the *Original* model as shown in Table 1. This experiment thus well answers Question 3 that we raised.

Graph Classification

In this section, we aim to answer Question 4 by evaluating the performance of variants of GAT-based GNN (GAT-GC) on graph classification benchmarks. Besides, we compare

Table 4: Testing accuracies(%) for graph classification. We highlight the result of the best performed model for each dataset. Our GAT-GC (f-Scaled) model achieves the top 2 on all 6 datasets.

	Datasets	MUTAG	PROTEINS	ENZYMES	NCI1	RE-B	RE-M5K
Baselines	WL	82.05 ± 0.36	74.68 ± 0.49	52.22 ± 1.26	82.19 ± 0.18	81.10 ± 1.90	49.44 ± 2.36
	PSCN	88.95 ± 4.37	75.00 ± 2.51	-	76.34 ± 1.68	86.30 ± 1.58	49.10 ± 0.70
	DGCNN	85.83 ± 1.66	75.54 ± 0.94	51.00 ± 7.29	74.44 ± 0.47	76.02 ± 1.73	48.70 ± 4.54
	GIN	89.40 ± 5.60	76.20 ± 2.80	-	$\mathbf{82.70 \pm 1.70}$	92.40 ± 2.50	$\mathbf{57.50 \pm 1.50}$
	CapsGNN	86.67 ± 6.88	76.28 ± 3.63	54.67 ± 5.67	78.35 ± 1.55	-	52.88 ± 1.48
	GAT-GC (f-Scaled)	$\mathbf{90.44 \pm 6.44}$	$\mathbf{76.81 \pm 3.77}$	$\mathbf{58.45 \pm 6.35}$	82.28 ± 1.81	$\mathbf{92.57 \pm 2.06}$	57.22 ± 2.20

our best-performed CPA model with baseline models to answer Question 5.

Social Network Datasets Since the RE-B and RE-M5K datasets don't have original node features and we assign all the node features to be the same, we have $P = 100.0\%$ in those datasets. Thus **all** multisets in aggregation will be mapped to the same embedding by the *Original* GAT-GC. After a mean readout function on all multisets, **all** graphs are finally mapped to the same embedding. The performance of the *Original* model is just random guessing of the graph labels as reported in Table 2. While our CPA models can distinguish all different multisets and are confirmed to be significantly better than the *Original* one.

Here we examine a naive approach to incorporate the cardinality information in the *Original* model by assigning node degrees as input node labels. By doing this way, the node features are diverse and we get $P = 0.0\%$, which means that the cases in Theorem 1 can be all avoided. However, the testing accuracies of *Original* can only reach $76.65 \pm 9.87\%$ on RE-B and $43.71 \pm 9.05\%$ on RE-M5K, which are significantly lower than the results of CPA models in Table 2. Thus in practice, our proposed models exhibit good generalization power comparing to the naive approach.

Bioinformatics Datasets For bioinformatics datasets that contain diverse node labels, we also report the P values in Table 3. The results reveal the existence ($P \geq 29.3\%$) of the cases in those datasets that can fool the *Original* model, thus the discriminative power of the *Original* model is *theoretically* constrained.

To empirically validate this, we compare the training accuracies of GAT-GC variants, since the discriminative power can be directly indicated by the accuracies on *training sets*. Higher training accuracy indicates a better fitting ability to distinguish different graphs. The training curves of GAT-GC variants are shown in Figure 2. From these curves, we can see even though the *Original* model has overfitted different datasets, the fitting accuracies that it converges to can never be higher than those of our CPA models. Compared to the WL kernel, CPA models can get training accuracies close to 100% on several datasets, which reach those obtained from the WL kernel (equal to 100% as shown in (Xu et al. 2019)). These findings validate that the discriminative power of the *Original* model is constrained while our CPA models can approach the upper bound of discriminative power with certain learned weights.

In Table 3 we report the testing accuracies of GAT-GC variants on bioinformatics datasets. The *Original* model can get meaningful results. However, we find our proposed CPA models further improve the testing accuracies of the *Original* model on all datasets. This indicates that the preservation of cardinality can also benefit the generalization power of the model besides the discriminative power.

From previous results in Table 2 and 3, we find the *f-Scaled* model performs the best with an average ranking measure (Taheri, Gimpel, and Berger-Wolf 2018). The good performance of the fixed-weight models (*f-Additive* and *f-Scaled*) comparing to the full models (*Additive* and *Scaled*) demonstrates that the improvements achieved by CPA models are not simply due to the increased capacities given by the additional vectors embedded.

Comparison to Baselines We further compare the best-performed GAT-GC variant (*f-Scaled*) with other baselines (WL kernel (WL) (Shervashidze et al. 2011), PATCHY-SAN (PSCN) (Niepert, Ahmed, and Kutzkov 2016), Deep Graph CNN (DGCNN) (Zhang et al. 2018), Graph Isomorphism Network (GIN) (Xu et al. 2019) and Capsule Graph Neural Network (CapsGNN) (Xinyi and Chen 2019)). In Table 4, we report the results. Our GAT-GC (f-Scaled) model achieves 4 top 1 and 2 top 2 on all 6 datasets. It is expected that even better performance can be achieved with certain choices of attention mechanism besides the GAT one.

Conclusion

In this paper, we theoretically analyze the representational power of GNNs with attention-based aggregators: We determine all cases when those GNNs always fail to distinguish distinct structures. The finding shows that the missing cardinality information in aggregation is the only reason to cause those failures. To improve, we propose cardinality preserved attention (CPA) models to solve this issue. In our experiments, we validate our theoretical analysis that the performances of the original attention-based GNNs are limited. With our models, the original models can be improved. Compared to other baselines, our best-performed model achieves competitive performance. In future work, a challenging problem is to better learn the attention weights so as to guarantee the injectivity of our cardinality preserved attention models after the training. Besides, it would be interesting to analyze the effects of different attention mechanisms.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Cai, J.-Y.; Fürer, M.; and Immerman, N. 1992. An optimal lower bound on the number of variables for graph identification. *Combinatorica* 12(4):389–410.

Duvenaud, D. K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; and Adams, R. P. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*, 2224–2232.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 1024–1034.

Hornik, K.; Stinchcombe, M.; and White, H. 1989. Multilayer feedforward networks are universal approximators. *Neural networks* 2(5):359–366.

Hornik, K. 1991. Approximation capabilities of multilayer feedforward networks. *Neural networks* 4(2):251–257.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 448–456.

Ivanov, S., and Burnaev, E. 2018. Anonymous walk embeddings. In *International Conference on Machine Learning*, 2191–2200.

Kingma, D. P., and Ba, J. 2018. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Kipf, T. N., and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

Knyazev, B.; Taylor, G. W.; and Amer, M. R. 2019. Understanding attention and generalization in graph neural networks. *arXiv preprint arXiv:1905.02850*.

Lee, J. B.; Rossi, R. A.; Kim, S.; Ahmed, N. K.; and Koh, E. 2018. Attention models in graphs: A survey. *arXiv preprint arXiv:1807.07984*.

Lee, J.; Lee, I.; and Kang, J. 2019. Self-attention graph pooling. In *International Conference on Machine Learning*, 3734–3743.

Li, Y.; Tarlow, D.; Brockschmidt, M.; and Zemel, R. 2016. Gated graph sequence neural networks. In *International Conference on Learning Representations*.

Li, G.; Müller, M.; Thabet, A.; and Ghanem, B. 2019. Deepgcns: Can gcns go as deep as cnns? In *The IEEE International Conference on Computer Vision (ICCV)*.

Maron, H.; Ben-Hamu, H.; Serviansky, H.; and Lipman, Y. 2019. Provably powerful graph networks. In *Advances in Neural Information Processing Systems*.

Morris, C.; Ritzert, M.; Fey, M.; Hamilton, W. L.; Lenssen, J. E.; Rattan, G.; and Grohe, M. 2019a. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of AAAI Conference on Artificial Inteligence*.

Morris, C.; Ritzert, M.; Fey, M.; Hamilton, W. L.; Lenssen, J. E.; Rattan, G.; and Grohe, M. 2019b. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 4602–4609.

Niepert, M.; Ahmed, M.; and Kutzkov, K. 2016. Learning convolutional neural networks for graphs. In *International conference on machine learning*, 2014–2023.

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

Shervashidze, N.; Schweitzer, P.; Leeuwen, E. J. v.; Mehlhorn, K.; and Borgwardt, K. M. 2011. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research* 12(Sep):2539–2561.

Taheri, A.; Gimpel, K.; and Berger-Wolf, T. 2018. Learning graph representations with recurrent neural network autoencoders. *KDD Deep Learning Day*.

Thekumparampil, K. K.; Wang, C.; Oh, S.; and Li, L.-J. 2018. Attention-based graph neural network for semi-supervised learning. *arXiv preprint arXiv:1803.03735*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

Weisfeiler, B., and Leman, A. 1968. The reduction of a graph to canonical form and the algebra which appears therein. *NTI, Series* 2.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Yu, P. S. 2019. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*.

Xinyi, Z., and Chen, L. 2019. Capsule graph neural network. In *International Conference on Learning Representations*.

Xu, K.; Li, C.; Tian, Y.; Sonobe, T.; Kawarabayashi, K.-i.; and Jegelka, S. 2018. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*, 5449–5458.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How powerful are graph neural networks? In *International Conference on Learning Representations*.

Ying, Z.; You, J.; Morris, C.; Ren, X.; Hamilton, W.; and Leskovec, J. 2018. Hierarchical graph representation learning with differentiable pooling. In *Advances in Neural Information Processing Systems*, 4805–4815.

Zhang, M.; Cui, Z.; Neumann, M.; and Chen, Y. 2018. An end-to-end deep learning architecture for graph classification. In *Proceedings of AAAI Conference on Artificial Inteligence*.

Zhou, J.; Cui, G.; Zhang, Z.; Yang, C.; Liu, Z.; and Sun, M. 2018. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*.

Proof for Lemma 1

Proof. Local-level: For the aggregator in the first layer, it will map different 1-height subtree structures to different embeddings from the distinct input multisets of neighborhood node features, since it's injective. Iteratively, the aggregator in the l-th layer can distinguish different l-height subtree structures by mapping them to different embeddings from the distinct input multisets of l-1-height subtree features, since it's injective.

Global level: From Lemma 2 and Theorem 3 in (Xu et al. 2019), we know: When all functions in $\mathcal{A}$ are injective, $\mathcal{A}$ can reach its upper bound of discriminative power, which is the same as the Weisfeiler-Lehman (WL) test (Weisfeiler and Leman 1968) when deciding the graph isomorphism. □

Proof for Theorem 1

Proof. To prove Theorem 1, we have consider both two directions in the iff statement:

(1) If given $c_1 = c_2 = c$, $X_1 = (S, \mu)$ and $X_2 = (S, k \cdot \mu)$, as $h(c, X) = \sum_{x \in X} \alpha_{cx} f(x)$, we have:

$$h(c_i, X_i) = \sum_{x \in X_i} \alpha_{cxi} f(x), i \in \{1, 2\},$$

where α_{cxi} is the attention weight belongs to X_i, and between $f(c)$ and $f(x)$, $x \in X_i, i \in \{1, 2\}$.

We can rewrite the equations using S and μ:

$$h(c_1, X_1) = h(c, S, \mu) = \sum_{s \in S} \mu(s) \alpha_{cs1} f(s),$$

$$h(c_2, X_2) = h(c, S, k \cdot \mu) = \sum_{s \in S} k \cdot \mu(s) \alpha_{cs2} f(s),$$

where $\mu(s)$ is the multiplicity function, and α_{csi} is the attention weight belongs to X_i, and between $f(c)$ and $f(s)$, $s \in S, i \in \{1, 2\}$.

Considering the softmax function in Equation 2 of our paper, we can use attention coefficient e to rewrite the equations:

$$\begin{aligned}\sum_{s \in S} \mu(s) \alpha_{cs1} f(s) &= \sum_{s \in S} \mu(s) \frac{\exp(e_{cs1})}{\sum_{x \in X_1} \exp(e_{cx1})} f(s) \\ &= \frac{\sum_{s \in S} \mu(s) \exp(e_{cs1})}{\sum_{x \in X_1} \exp(e_{cx1})} f(s),\end{aligned}$$

$$\begin{aligned}\sum_{s \in S} k \cdot \mu(s) \alpha_{cs2} f(s) &= k \cdot \sum_{s \in S} \mu(s) \frac{\exp(e_{cs2})}{\sum_{x \in X_2} \exp(e_{cx2})} f(s) \\ &= k \cdot \frac{\sum_{s \in S} \mu(s) \exp(e_{cs2})}{\sum_{x \in X_2} \exp(e_{cx2})} f(s),\end{aligned}$$

where e_{csi} is the attention coefficient belongs to X_i, and between $f(c)$ and $f(s)$, $s \in S, i \in \{1, 2\}$. Moreover, e_{cxi} is the attention coefficient belongs to X_i, and between $f(c)$ and $f(x)$, $x \in X_i, i \in \{1, 2\}$.

As attention coefficient e is computed by function Att, which is regardless of X, thus $e_{cs1} = e_{cs2}$, $\forall s \in S$ and $e_{cx1} = e_{cx2}$, $\forall x \in X_1, X_2$. We denote $e_{cx} = e_{cx1} = e_{cx2}$, $e_{cs} = e_{cs1} = e_{cs2}$. Remind that X_2 has k copies of the elements in X_1, so that

$$\sum_{x \in X_1} \exp(e_{cx}) = \frac{1}{k} \sum_{x \in X_2} \exp(e_{cx}).$$

Using this equation, we can get

$$\begin{aligned}\frac{\sum_{s \in S} \mu(s) \exp(e_{cs1})}{\sum_{x \in X_1} \exp(e_{cx1})} f(s) &= \frac{\sum_{s \in S} \mu(s) \exp(e_{cs})}{\frac{1}{k} \sum_{x \in X_2} \exp(e_{cx})} f(s) \\ &= k \cdot \frac{\sum_{s \in S} \mu(s) \exp(e_{cs2})}{\sum_{x \in X_2} \exp(e_{cx2})} f(s).\end{aligned}$$

From all equations above, we finally have

$$\begin{aligned}h(c_1, X_1) &= \frac{\sum_{s \in S} \mu(s) \exp(e_{cs1})}{\sum_{x \in X_1} \exp(e_{cx1})} f(s) \\ &= k \cdot \frac{\sum_{s \in S} \mu(s) \exp(e_{cs2})}{\sum_{x \in X_2} \exp(e_{cx2})} f(s) \\ &= h(c_2, X_2).\end{aligned}$$

(2) If given $h(c_1, X_1) = h(c_2, X_2)$ for all f, Att, we have

$$\sum_{x \in X_1} \alpha_{cx1} f(x) = \sum_{x \in X_2} \alpha_{cx2} f(x), \quad \forall f, Att,$$

where α_{cxi} is the attention weight belongs to X_i, and between $f(c_i)$ and $f(x)$, $x \in X_i, i \in \{1, 2\}$.

We denote $X_1 = (S_1, \mu_1)$ and $X_2 = (S_2, \mu_2)$ and rewrite the equation:

$$\sum_{s \in S_1} \mu_1(s) \alpha_{cs1} f(s) = \sum_{s_\in S_2} \mu_2(s) \alpha_{cs2} f(s), \quad \forall f, Att,$$

where $\mu_i(s)$ is the multiplicity function of $X_i, i \in \{1, 2\}$. Moreover, α_{csi} is the attention weight belongs to X_i, and between $f(c_i)$ and $f(s)$, $s \in S_i, i \in \{1, 2\}$.

When considering the relations between S_1 and S_2, we have:

$$\begin{aligned}&\sum_{s \in S_1 \cap S_2} \big(\mu_1(s) \alpha_{cs1} - \mu_2(s) \alpha_{cs2}\big) f(s) + \\ &\sum_{s \in S_1 \setminus S_2} \mu_1(s) \alpha_{cs1} f(s) - \sum_{s \in S_2 \setminus S_1} \mu_2(s) \alpha_{cs2} f(s) = 0. \quad (8)\end{aligned}$$

If we assume the equality of Equation 8 is true for all f and $S_1 \neq S_2$, we can define such two functions f_1 and f_2:

$$\begin{aligned}f_1(s) &= f_2(s), && \forall s \in S_1 \cap S_2, \\ f_1(s) &= f_2(s) - 1, && \forall s \in S_1 \setminus S_2, \\ f_1(s) &= f_2(s) + 1, && \forall s \in S_2 \setminus S_1.\end{aligned}$$

If given the equality of Equation 8 is true for f_1, we have:

$$\begin{aligned}&\sum_{s \in S_1 \cap S_2} \big(\mu_1(s) \alpha_{cs1} - \mu_2(s) \alpha_{cs2}\big) f_1(s) + \\ &\sum_{s \in S_1 \setminus S_2} \mu_1(s) \alpha_{cs1} f_1(s) - \sum_{s \in S_2 \setminus S_1} \mu_2(s) \alpha_{cs2} f_1(s) = 0.\end{aligned} \quad (9)$$

We can rewrite Equation 9 using f_2:

$$\sum_{s\in S_1\cap S_2}\big(\mu_1(s)\alpha_{cs1}-\mu_2(s)\alpha_{cs2}\big)f_2(s)+$$
$$\sum_{s\in S_1\setminus S_2}\mu_1(s)\alpha_{cs1}(f_2(s)-1)-$$
$$\sum_{s\in S_2\setminus S_1}\mu_2(s)\alpha_{cs2}(f_2(s)+1)=0.$$

Thus we know

$$\sum_{s\in S_1\cap S_2}\big(\mu_1(s)\alpha_{cs1}-\mu_2(s)\alpha_{cs2}\big)f_2(s)+$$
$$\sum_{s\in S_1\setminus S_2}\mu_1(s)\alpha_{cs1}f_2(s)-\sum_{s\in S_2\setminus S_1}\mu_2(s)\alpha_{cs2}f_2(s)=$$
$$\sum_{s\in S_1\setminus S_2}\mu_1(s)\alpha_{cs1}+\sum_{s\in S_2\setminus S_1}\mu_2(s)\alpha_{cs2} \tag{10}$$

Note that the LHS of Equation 10 is just the LHS of Equation 8 when $f=f_2$. As $\mu_i(s)\geq 1$ due to the definition of multiplicity, $\alpha_{csi}>0$ due to the softmax function, we have $\mu_i(s)\alpha_{csi}>0, \forall s\in S_i, i\in\{1,2\}$. Thus the RHS of Equation 10 > 0 and we now know the equality in Equation 8 is not true for f_2. So the assumption of $S_1\neq S_2$ is false.

We denote $S=S_1=S_2$. To let the remaining summation term always equal to 0, we have

$$\mu_1(s)\alpha_{cs1}-\mu_2(s)\alpha_{cs2}=0,\quad \forall Att.$$

Considering Equation 2 in our paper, we can rewrite the equation above:

$$\frac{\mu_1(s)}{\mu_2(s)}=\frac{\exp(e_{cs2})}{\exp(e_{cs1})}\frac{\sum_{x\in X_1}\exp(e_{cx1})}{\sum_{x\in X_2}\exp(e_{cx2})},\quad \forall Att, \tag{11}$$

where e_{csi} is the attention coefficient belongs to X_i, and between $f(c_i)$ and $f(s)$, $s\in S$. And e_{cxi} is the attention coefficient belongs to X_i, and between $f(c_i)$ and $f(x)$, $x\in X_i, i\in\{1,2\}$.

The LHS of Equation 11 is a rational number. However if $c_1\neq c_2$, the RHS of Equation 11 can be irrational: We assume S contains at least two elements s_0 and $s\neq s_0$. If not, we can directly get $c_1=c_2$. We consider any attention mechanism that results in:

$$e_{cs1}=1,\ \forall s\in S,$$
$$e_{cs2}=\begin{cases}1, & \text{for} s=s_0,\\ 2, & \forall s\neq s_0\in S.\end{cases}$$

Thus when $s=s_0$, the RHS of the equation become:

$$\frac{e}{e}\frac{|X_1|e}{(|X_2|-n)e^2+ne}=\frac{|X_1|}{(|X_2|-n)e+n},$$

where n is the multiplicity of s_0 in X_2. It is obvious that the value of RHS is irrational. So we have $c_1=c_2$ to always hold the equality.

With $c_1=c_2$, we know $e_{cs1}=e_{cs2}, \forall s\in S$ and $e_{cx1}=e_{cx2}, \forall x\in X_1, X_2$. We denote $e_{cx}=e_{cx1}=e_{cx2}$, Equation 11 becomes

$$\frac{\mu_1(s)}{\mu_2(s)}=\frac{\sum_{x\in X_1}\exp(e_{cx})}{\sum_{x\in X_2}\exp(e_{cx})}=const.,\quad \forall Att.$$

We further denote $k=\mu_1(s)/\mu_2(s), \forall s\in S$. So that $\mu_2=k\cdot\mu_1$. Finally by denoting $\mu=\mu_1$, we have $X_1=(S,\mu)$, $X_2=(S,k\cdot\mu)$ and $c_1=c_2$. □

Proof for Corollary 1

Proof. For subtrees, if S_1 and S_2 are 1-height subtrees that have the same root node feature and the same distribution of node features, $\mathcal{A}$ will get the same embeddings for S_1 and S_2 according to Theorem 1.

For graphs, let G_1 be a fully connect graph with n nodes and G_2 be a ring-like graph with n nodes. All nodes in G_1 and G_2 have the same feature x. It is clear that the Weisfeiler-Lehman test of isomorphism decides G_1 and G_2 as non-isomorphic.

We denote $\{X_i\}, i\in G_1$ as the set of multisets for aggregation in G_1, and $\{X_j\}, j\in G_2$ as the set of multisets for aggregation in G_2. As G_1 is a fully connect graph, all multisets in G_1 contain 1 central node and $n-1$ neighbors. As G_2 is a ring-like graph, all multisets in G_2 contain 1 central node and 2 neighbors. Thus we have

$$X_i=(\{x\},\{\mu_1(x)=n\}),\ \forall i\in G_1,$$
$$X_j=(\{x\},\{\mu_2(x)=3\}),\ \forall j\in G_2,$$

where $\mu_i(x)$ is the multiplicity function of the node with feature x in $G_i, i\in\{1,2\}$.

From Theorem 1, we know that $h(c,X_i)=h(c,X_j), \forall i\in G_1, \forall j\in G_2$. Considering the Equation 3 of our paper, we have $h_i^l=h_j^l, \forall i\in G_1, \forall j\in G_2$ in each iteration l. Besides, as the number of node in G_1 and G_2 are equals to n, $\mathcal{A}$ will always map G_1 and G_2 to the same set of multisets of node features $\{h^l\}$ in each iteration l and finally get the same embedding for each graph. □

Proof for Corollary 2

Proof. Given two distinct multiset of node features X_1 and X_2 that have the same central node feature and the same distribution of node features: $c_1=c_2$, $X_1=(S,\mu)$ and $X_2=(S,k\cdot\mu)$ for $k\in\mathbb{N}^*$, we know the cardinality of X_2 is k times of the cardinality of X_1. Thus X_1 and X_2 can be distinguished by their cardinality.

However, the weighted summation function h in attention-based aggregator $\mathcal{A}$ will map them to the same embedding: $h(c_1,X_1)=h(c_2,X_2)$ according to Theorem 1. Thus we cannot distinguish X_1 and X_2 via $\mathcal{A}$. To conclude, $\mathcal{A}$ lost the cardinality information after aggregation. □

Proof for Corollary 3

Proof. For any two distinct multisets X_1 and X_2 that $\mathcal{T}$ previously always fail to distinguish according to Theorem 1, we denote $X_1=(S,\mu)$ and $X_2=(S,k\cdot\mu)\subset\mathcal{X}$ for some $k\in\mathbb{N}^*$ and $c\in S$. Thus $\sum_{x\in X_1}\alpha_{cx1}f(x)=\sum_{x\in X_2}\alpha_{cx2}f(x)$, where α_{cxi} is the attention weight belongs to X_i, and between $f(c)$ and $f(x), x\in X_i, i\in\{1,2\}$. We denote $H=\sum_{x\in X_1}\alpha_{cx1}f(x)=\sum_{x\in X_2}\alpha_{cx2}f(x)$. When applying CPA models, the aggregation functions in $\mathcal{T}$

can be rewritten as:

$$h_1(c, X_i) = H + w \odot \sum\nolimits_{x \in X_i} f(x), \quad i \in \{1, 2\},$$
$$h_2(c, X_i) = \psi(\left|X_i\right|) \odot H, \quad i \in \{1, 2\}.$$

We consider the following example: All elements in w equal to 1. Function ψ maps $\left|X\right|$ to a n-dimensional vector which all elements in it equal to $\left|X\right|$. And $f(x) = N^{-Z(x)}$, where $Z : \mathcal{X} \to \mathbb{N}$ and $N > \left|X\right|$. So that the aggregation functions become:

$$h_1(c, X_i) = H + \sum\nolimits_{x \in X_i} f(x), \quad i \in \{1, 2\},$$
$$h_2(c, X_i) = \left|X_i\right| \cdot H, \quad i \in \{1, 2\}.$$

For h_1, we have $h_1(c, X_1) - h_1(c, X_2) = \sum_{x \in X_1} f(x) - \sum_{x \in X_2} f(x)$. According to Lemma 5 of (Xu et al. 2019), when $X_1 \neq X_2$, $\sum_{x \in X_1} f(x) \neq \sum_{x \in X_2} f(x)$. So $h_1(c, X_1) \neq h_1(c, X_2)$.

For h_2, we have $h_2(c, X_1) - h_2(c, X_2) = (\left|X_1\right| - \left|X_2\right|) \cdot H$. As $\alpha_{cx} > 0$ due to the softmax function, and $f(x) > 0$ in our example, we know $H > 0$. Moreover as $\left|X_1\right| - \left|X_2\right| \neq 0$, we can get $h_2(c, X_1) \neq h_2(c, X_2)$. □

Details of Datasets

For the node classification task, we generate a graph with 4800 nodes and 32400 edges. 40.58% of the nodes are included in triangles as vertices while 59.42% are not. There are 4000 nodes assigned with feature '0', 400 with feature '1' and 400 with feature '2'. The label of each node for prediction is whether or not it's included in a triangle.

For the graph classification task, detailed statistics of the bioinformatics and social network datasets are listed in Table 5. All of the datasets are available at https://ls11-www.cs.tu-dortmund.de/staff/morris/graphkerneldatasets.

In all datasets, if the original node features are provided, we use the one-hot encodings of them as input.

Table 5: Dataset Description

Datasets	Graphs	Classes	Features	Node Avg.	Edge Avg.
MUTAG	188	2	7	17.93	19.79
PROTEINS	1113	2	4	39.06	72.81
ENZYMES	600	6	6	32.63	62.14
NCI1	4110	2	23	29.87	32.30
RE-B	2000	2	-	429.63	995.51
RE-M5K	4999	5	-	508.52	1189.75

Details of Experiment Settings

For all experiments, we perform 10-fold cross-validation and repeat the experiments 10 times for each dataset and each model. To get a final accuracy for each run, we select the epoch with the best cross-validation accuracy averaged over all 10 folds. The average accuracies and their standard deviations are reported based on the results across the folds in all runs.

In our *Additive* and *Scaled* models, all MLPs have 2 layers with ReLU activation.

In the GAT variants, we use 2 GNN layers and a hidden dimensionality of 32. The negative input slope of $\mathrm{LeakyReLU}$ in the GAT attention mechanism is 0.2. The number of heads in multi-head attention is 1.

In the GAT-GC variants, we use 4 GNN layers. For the $\mathrm{Readout}$ function in all models, we use sum for bioinformatics datasets and mean for social network datasets. We apply Batch normalization (Ioffe and Szegedy 2015) after every hidden layers. The hidden dimensionality is set as 32 for bioinformatics datasets and 64 for social network datasets. The negative input slope of $\mathrm{LeakyReLU}$ in the GAT attention mechanism is 0.2. We use a single head in the multi-head attention in all models.

All models are trained using the Adam optimizer (Kingma and Ba 2018) and the learning rate is dropped by a factor of 0.5 every 400 epochs in the node classification task and every 50 epochs in the graph classification task. We use an initial learning rate of 0.01 for the TRIANGLE-NODE and bioinformatics datasets and 0.0025 for the social network datasets. For the GAT variants, we use a dropout ratio of 0 and a weight decay value of 0. For the GAT-GC variants on each dataset, the following hyper-parameters are tuned: (1) Batch size in $\{32, 128\}$; (2) Dropout ratio in $\{0, 0.5\}$ after dense layer; (3) L_2 regularization from 0 to 0.001. On each dataset, we use the same hyper-parameter configurations in all model variants for a fair comparison.

For the results of the baselines for comparison, we use the results reported in their original works by default. If results are not available, we use the best testing results reported in (Xinyi and Chen 2019; Ivanov and Burnaev 2018).