GraTeD-MLP: Efficient Node Classification via Graph Transformer Distillation to MLP

Sarthak Malik[*]
AI Garage, Mastercard, India

Aditi Rai[*]
AI Garage, Mastercard, India

Ram Ganesh V[*]
AI Garage, Mastercard, India

Himank Sehgal
AI Garage, Mastercard, India

Akshay Sethi
AI Garage, Mastercard, India

Aakarsh Malhotra[†]
AI Garage, Mastercard, India

Abstract

Graph Transformers (GTs) like NAGphormer have shown impressive performance by encoding graph's structural information and node features. However, their self-attention and complex architectures require high computation and memory, hindering their deployment. Thus, we propose a novel framework called **Gra**ph **T**ransform**e**r **D**istillation to **M**ulti-**L**ayer **P**erceptron (GraTeD-MLP). GraTeD-MLP leverages knowledge distillation (KD) and a novel decomposition of attentional representation to distill the learned representations from the teacher GT to a student MLP. During distillation, we incorporate a multi branch MLP architecture where two branches learn the decomposed attentional representation for a node while the third predicts node embeddings. Encoding the attentional representation mitigates the MLP's over-reliance on node features, enabling robust performance even in inductive settings. Empirical results demonstrate that the proposed GraTeD-MLP has significantly faster inference time than the teacher GT model, with speed-up ranging from $20\times - 40\times$. With up to 25% improved performance over vanilla MLP. Furthermore, we empirically show that the proposed GraTeD-MLP outperforms other GNN distillation methods in seven datasets in both inductive and transductive settings.

1 Introduction

Graphs are a robust data structure that models real-world relationships and interactions. From social networks [1] to biological protein interactions [2], graphs provide an intuitive framework to represent complex interconnected systems. Graph Neural Networks (GNNs) emerged as a significant advancement, using message-passing mechanisms to aggregate information from neighboring nodes and learn rich representations. Despite their success, GNNs [3, 4] still face challenges in capturing long-range dependencies and global graph structure. Subsequently, Graph Transformers (GT) [5–7] emerged as a groundbreaking approach. GTs use a self-attention mechanism to elevate graph representation learning with its expressiveness capabilities, which allows them to capture long-range dependencies and global contexts. Leveraging attention mechanisms also allows GTs to alleviate the over smoothing problem [8, 9] present in GNNs. Consequently, GTs have demonstrated superior performance over GNNs across node and graph classification tasks. However, the computational complexity and memory requirements of GTs pose significant scalability challenges.

The complex architectures and heavy reliance on self-attention mechanisms make GTs computationally demanding and resource-intensive. Furthermore, these GT methods treat all the nodes in the graph as independent tokens. This results in higher latency, limiting their applicability to large-scale graphs and resource-constrained environments to a greater extent than GNNs. As a solution, NAGphormer [10] treated nodes as independent sequences, which enabled mini-batch training and had the

[*]Equal contribution.

[†]Corresponding Author, email: aakarsh.malhotra@mastercard.com

Malik et al., GraTeD-MLP: Efficient Node Classification via Graph Transformer Distillation to MLP. *Proceedings of the Third Learning on Graphs Conference (LoG 2024)*, PMLR XXX, Virtual Event, November 26–29, 2024.

potential to handle larger graphs, unlike previous architectures. Despite the advantages in training, NAGphormer still requires access to the graph and the associated embeddings during inference, which hinders its deployment in real-world applications.

To mitigate such issues, research efforts have focused on graph knowledge distillation (KD). KD aims to transfer the expressive power of a teacher GNN into a lightweight surrogate student model [11]. These student models mimic teacher GNN's behavior while exhibiting superior computational efficiency and scalability. The mimicking is performed using hint-based methods [12, 13], which aims to reduce KL divergence between teacher and student soft-label representation. In graphs, the student model could be a GNN (or a variant of GNNs, such as Graph Convolutional Networks / Graph Attention Networks) or a Multi-Layer Perceptron (MLPs). While the former GNN would be lighter than a teacher, it would still require test-time graphs, making it infeasible in large-scale applications. On the contrary, the latter MLP student model can be trained to approximate the decision boundaries of GNNs through KD frameworks without retrieving graph topology and neighborhood feature information.

GCN-to-GCN and GCN-to-MLP distillation are well-established in the literature and are known to be effective. Their effectiveness of knowledge "transferability" can be seen by Centered Kernel Alignment (CKA)[3] metric [14]. CKA compares hidden dimension embedding $H \in R^{n \times d_1}$ and $\hat{H} \in R^{n \times d_2}$ for two heterogeneous models. The metric CKA $\in [0, 1]$, with 1 denoting complete transferability. A higher CKA would imply a more effective alignment between the feature spaces of the teacher and the student model. As seen in Figure 1, transferability is higher and around 0.8 in the case of GCN-GCN and a GCN-MLP knowledge distillation.

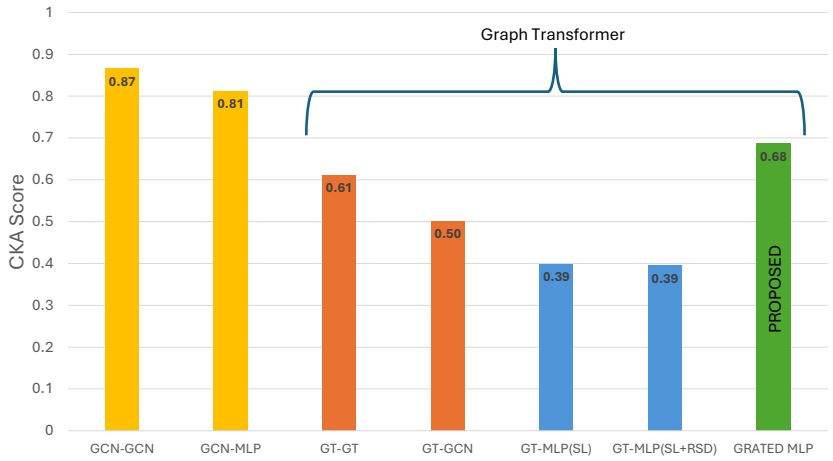


Figure 1: Knowledge transferability in distillation across models.

Though the transferability of GT-to-GT distillation is still somewhat higher at 0.61, we observe that the transferability drops to 0.5 when distilling the GT to a GCN student (we use NAGphormer as a GT here). It drops to 0.4 when distilling NAGphormer (GT) to an MLP through existing hint-based distillation methods. While the former GT-to-GCN is a cross-architecture distillation, the latter (GT-to-MLP) aggravates the problem by being cross-architecture and lacks graph structure. While GCNs can still somewhat capture the graph structure using a message-passing mechanism, an MLP lacks a message-passing function and self-attention, which empowers a GT to encode graph structural knowledge. Thus, we require a specialized GT-to-MLP distillation framework to effectively harness the power of attention and the node features. This way, we can achieve improved performance due to attention while utilizing the fast inference capabilities of an MLP. However, no such work exists in the GT literature due to GTs' inherent scalability limitations and higher latency than GNNs, as described in the forthcoming section.

As a solution, this work proposes a novel distillation paradigm that leverages a multi branch MLP network to distill knowledge from a pre-trained GT. The MLP architecture comprises three distinct branches: one is dedicated to predicting the output predictions by taking node features as input. In

[3]details in Appendix A.1

contrast, the other two branches estimate the attention scores and intermediate layer representations of the GT. The final prediction is obtained by merging the three branches. By jointly optimizing the two branches during the distillation, MLP captures the attention. Thus, the main contributions of the proposed work are as follows:

1. GraTeD-MLP, a novel distillation framework that distills GT into a lightweight, efficient MLP.
2. Extensive evaluation of the GraTeD-MLP on seven datasets for node classification under transductive and inductive settings.
3. Significant speed-ups ranging from 20× to 40× faster inference times across various datasets, upto 25% improvement over vanilla MLP, and better knowledge transferability, as illustrated by CKA score.

2 Literature Review

While graph partitioning and parallel training is widely explore in literature [15–18], it still needs to keep the graph during inference. Knowledge distillation [11] has been widely used to transfer knowledge from a large teacher into a smaller student model. In this section, we first summarize graph transformers and showcase existing methods for KD in a cross-architecture setting, highlighting the gap in distillation from GT-to-MLP. Subsequently, we review methods of graph KD and how they have been modified for a fair comparison for this study.

2.1 Graph Transformers and Cross Architecture Distillation

Transformer-based architectures have been successful in various domains [19–21]. However, since graph data typically contains structural information, it cannot be directly represented as tokens. Existing works [5, 22] leverage various structural encoding techniques like eigenvectors of graph laplacian and an invariant aggregation of laplacian eigenvectors of positional encoding. Most existing GTs treat nodes as independent tokens, creating a quadratic complexity O(n^2) during training. Also, though GTs have shown exceptional ability in graph classification, performance in node classification is sub-optimal [23]. This is because the attention mechanism focuses on the entire graph, resulting in an over-globalizing problem. To fix this issue, NAGphormer [10] proposed to treat each node and its hops as a sequence and construct that sequence by aggregating all node representations for a particular hop. The NAGphormer can scale up to relatively larger graphs by treating nodes as individual sequences. Furthermore, by creating specific sequences for nodes, the NAGphormer can better leverage node-specific local information for node classification tasks.

To further reduce inference time and utilize the power of GTs like NAGphormer, the aim is to distill NAGphormer to MLP. However, most KD studies [11, 24–30] have focused on distilling in a homogenous framework, i.e., the teacher and student have similar architecture. In such a homogeneous setting, student and teacher models have similar feature spaces and methods. Thus, straightforward methods like mimicking probability distributions and variations of representational similarity (RSD) loss can achieve feature space alignment. On the other hand, cross-architecture distillation is relatively unexplored, making it harder to create suitable student models. For instance, Liu et al. [31] and Hao et al. [14] demonstrated this lack of transferability in vision datasets using the CKA and cosine similarity metrics. Consequently, they developed novel frameworks to bridge this gap by aligning the intermediate feature spaces. In the vision domain, various models can access the entire image view. However, an MLP lacks access to the whole graph structure at inference. Thus, there is a need to develop a specialized framework for distilling graph transformers to MLP.

2.2 Graph Knowledge Distillation

Graph-based models are inherently complex and rely on complex message-passing structures. Various studies [32–36] aim to make more efficient student models. These studies have focused on developing smaller GNNs comparable to the teacher, although the reliance on message passing still results in latency issues. To alleviate this issue, recent works [12, 13, 37–40] have distilled structural insights into an MLP. These lightweight MLP models do not require graphs during inference time, resulting in significant speed improvements with comparable performance. GLNN [12] attempts to mimic the predictions of GNNs using soft label mechanism [11]. NOSMOG [13] incorporates structural information through an RSD loss. FF-G2M [39] leverages insights from spectral graph theory to

account for the loss of high-frequency information during distillation. In contrast, Cold Brew [37] addresses the cold start problem by learning a latent node embedding.

To the best of our knowledge, this is the first work on distilling a graph transformer into an MLP. To evaluate our proposed method, we modify existing GNN-to-MLP distillation benchmarks from GLNN [12] and NOSMOG [13]. While these benchmarks use a GNN as the teacher model, we replace the teacher with a Graph Transformer and utilize the soft label function from [12]. We also adopt the RSD loss from [13] for another benchmark. Additionally, distillation methods like Cold Brew [37] and FF-G2M [39] are based on GNN-specific principles such as message passing and neighborhood aggregation. They do not align with the global context attention-based architecture of a GT and are, therefore, not applicable to our setting.

3 Preliminaries

3.1 Notations

Let $G = (V, E)$ be a graph, where V, E, and n are vertices, edges, and size of the node-set, respectively. Let the $d-$dimensional feature vector of node $v \in V$ be $x_v \in X$. Here $X \in R^{n \times d}$ is the feature matrix characterizing the nodes. $A \in R^{n \times n}$ represents the adjacency matrix and let D be the diagonal matrix. For node classification, the task is to predict the probability of every node $v \in V$. Let $Y \in R^{n \times c}$ correspond to the ground truth category, with c being the number of classes. The labeled nodes are denoted by superscript ^{L}, and the unlabelled nodes are denoted by superscript ^{U}.

3.2 Transformer Preliminary

The most critical component of the transformer is the multi-head attention layer [41]. Let $X \in R^{n*d}$ be the input to the attention layer with n as the number of tokens. For NAGphormer [10], number of tokens corresponds to $(k+1)$, which is the number of hops in addition to the node. The Q, K,$V \in R^{d \times d/m}$ correspond to the query, key and value vectors and d is the hidden size and m is the number of heads. The final attention output matrix is given as:

$$A = softmax\left(\frac{QK^T}{\sqrt{d_K}}\right)V \tag{1}$$

$$Z = concat(A^1, A^2 ... A^m)W_z + b_z \tag{2}$$

4 GraTeD-MLP: Graph Transformer Distillation to MLP

As illustrated in Figure 2, we now propose a novel framework called GraTeD-MLP: **Gra**ph **T**ransform**e**r **D**istillation to **M**ulti-**L**ayer **P**erceptron. GraTeD-MLP uses low-rank property and decomposition of the attention matrix learned by the GT, enabling efficient distillation into a mulit branch MLP.

4.1 Attention Matrix Decomposition

The attention matrix A_i computed by the GT at the ith attention head has an inherent low-rank property, with an upper bound on its rank given by d/m, where d is the hidden size and m is the number of attention heads. This property stems from the rank theory of matrix products, which states that the rank of a matrix product $\leq$ is the minimum rank of the individual matrices:

$$R(A^i) \leq \min(R(Q^i), R(K^i), R(V^i)) \tag{3}$$

where R is the rank of the matrix, and Q^i, K^i, and V^i are the query, key, and value matrices, respectively, at the ith attention head. The low-rank property of the attention matrix A allows us to decompose it into its constituent components using Singular Value Decomposition (SVD) as $A = U\Sigma V^T$. Here U and V are orthogonal matrices representing the left and right singular vectors, respectively, and Σ is a diagonal matrix containing the singular values. The rank of A is equal to the number of non-zero singular values in Σ. Using this decomposition, we can distill the complex attention matrix into simpler components (U and V), which the student MLP model can learn more efficiently. This decomposition lets us capture the underlying structural information encoded within the attention matrix, facilitating efficient knowledge transfer from the GT to the MLP.

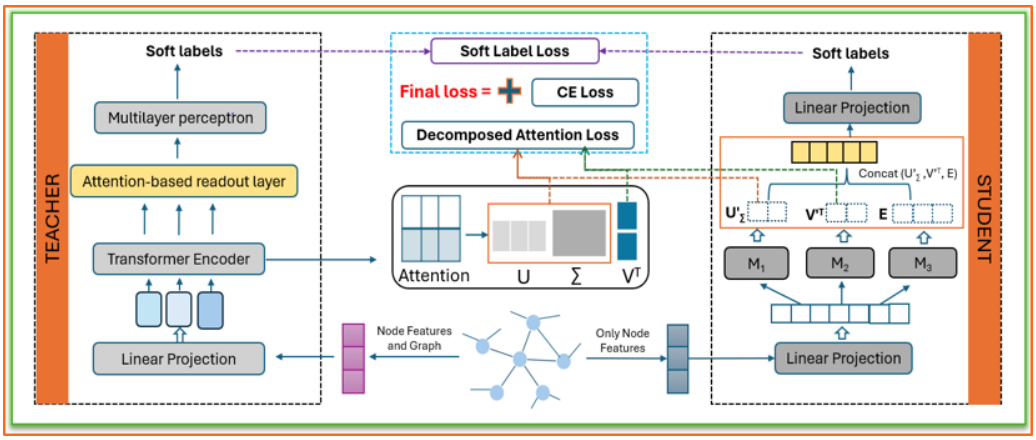


Figure 2: Overview of the proposed **Gra**ph **T**ransform**er** **D**istillation to **M**ulti-**L**ayer **P**erceptron (GraTeD-MLP) framework.

4.2 Dedicated Student for Attention and Features

Due to the inherent differences in the feature spaces between a vanilla MLP and GT, it is crucial to explicitly align the feature spaces during distillation. We employ a three-branch MLP architecture that learns to capture the structural and feature-level information separately. Thus, two student branches focus on KD for structure, while the third focuses on features.

1. Structure Student: The graph's structural information is encoded in the U and V singular vectors, as obtained from the SVD decomposition. The U vector represents an eigenbasis for the feature space of the nodes and the hop embeddings obtained from the Hop2 token algorithm. Each column in U represents a basis vector that captures a specific pattern of attention across the input space (nodes and hops). V captures how these patterns (from U) map back to the original attention scores. Each row in V represents a weight vector for one of the attention dimensions, showing how much each "direction" or pattern contributes to the original attention matrix. Hence, the V vector captures the relationships between different hops in the graph, encoding the structural dependencies and interactions. The two structure student branches of the multi branch MLP take the U and V vectors as input and learn to predict the corresponding components of the GT's attention matrix. By explicitly distilling U and V into separate branches of the MLP, we enable the student model to reconstruct and learn the attention matrix, thus transferring the structural insights from the GT to the MLP.

$$\mathcal{L}_{\text{structure}} = \lambda_3 \mathcal{L}_{U\Sigma} + \lambda_4 \mathcal{L}_V \tag{4}$$

Here, the two components of decomposed attention-based loss $\mathcal{L}_{\text{U}\Sigma}$ and $\mathcal{L}_{\text{V}}$ explicitly ensures that the model learns the structural information.

2. Feature Student: The third branch of GraTeD-MLP takes the node features as input and directly predicts the output embedding E. This branch is responsible for learning rich representation from the node's feature-level information, thus complementing the structural knowledge. With a series of fully connected layers that operate on the node features, this MLP learns to map the node features to the desired space. This is sufficient as the feature spaces of both models align as per the CKA metric shown above. The feature information is implicitly learned using the soft label loss, as explained below.

3. Traditional Soft Label Distillation Loss: As developed by Hinton et al. [11], the core idea of distillation is to develop a student model which can perform comparative to the complex teacher model. Let the ground truth label be y_v for any node in the labelled set $v \in V^L$ and let z_v be the soft target generated by the teacher NAGphormer. The objective function can then be formulated as:

$$\mathcal{L}_{\text{distill}} = \lambda_1 \mathcal{L}_{\text{CE}} + \lambda_2 \mathcal{L}_{\text{SL}} \tag{5}$$

where, $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss computed on the labeled nodes, ensuring that the model's predictions align with the ground truth labels. $\mathcal{L}_{\text{SL}}$ is a soft label based distillation loss for the structure student and feature student, respectively, encouraging their predictions to match those of the

GT teacher model. It is formulated using the Kullback-Leibler (KL) divergence between the student branches' predictions and the corresponding outputs of the GT teacher model, as elaborate below:

$$\mathcal{L}_{\text{distill}} = \lambda_1 \Sigma_{v \in V^L} L_{\text{CE}}(\hat{y_v}, y_v) + \lambda_2 \Sigma_{v \in V} KL(\hat{y_v}, z_v) \tag{6}$$

4.3 Fusion and Optimization

The outputs of the two structure and one feature students are fused to obtain the final predictions through a gating mechanism. The gating method adaptively combines the information from all three branches, allowing the model to adjust the relative importance of structural and feature-level information based on the input data. Furthermore, the branches of the MLP are jointly optimized using a combination of supervised cross-entropy loss and consistency losses to align their predictions with the GT teacher model. The overall optimization objective is formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{distill}} + \mathcal{L}_{\text{structure}} \tag{7}$$

which translates to:

$$\mathcal{L} = (\lambda_1 \mathcal{L}_{\text{CE}} + \lambda_2 \mathcal{L}_{\text{SL}}) + (\lambda_3 \mathcal{L}_{U\Sigma} + \lambda_4 \mathcal{L}_V) \tag{8}$$

The hyperparameters λ_1 to λ_4 control the relative importance of each loss and tune the desired trade-off between supervised learning and KD. By optimizing this objective function, the proposed GraTeD-MLP method distills the knowledge from the GT teacher, capturing both structural and feature-level information. With the student model, we obtain superior computational efficiency and scalability. The complete summary of the proposed method GraTeD-MLP is summarized in Algorithm 1 of the Appendix.

5 Experimental Details

Databases: To show the efficacy of the proposed GraTeD-MLP method, we showcase experiments on seven publically available datasets for node classification task: Cora [42], Pubmed [42], Photo [43], Computer [43], Citeeser [42], Aminer-CS [44], and OGB-Arxiv [45]. For the five datasets, we follow the data splits used in [12, 33], and for the large scale, we use the splits used in the respective paper.

Experimental Setting - Transductive vs. Inductive: The NAGphormer [10] paper was only evaluated on transductive settings. For the transductive setting, we train the model on the entire Graph G,X^L,Y^L and evaluate it on X^U and Y^U. Similar to [12, 13], all the nodes in the graph are used to generate soft labels. Furthermore, inductive settings are critical in the real world. From the unlabelled vertices V^U, we select V^U_{in}, which partitions it into two disjoint set of vertices $V^U = V^U_{obs} \cup V^U_{in}$. All the nodes belonging to V^U_{in} and its corresponding edges are held out, leading to two disjoint graphs $G = G_{obs} \cup G_{ind}$. Correspondingly, the labels and the node features are partitioned into three disjoint sets $Y = Y^L \cup Y^U_{obs} \cup Y^U_{in}$ and $X = X^L \cup X^U_{obs} \cup X^U_{in}$.

Comparative Algorithms: For baselines, we compare GraTeD-MLP with (1) teacher NAGphormer [10], (2) vanilla MLP, and current hint-based existing distillation methods, including (3) GLNN [12] inspired soft label (SL) student MLP, and (4) NOSMOG [13] inspired soft label + relational similarity distillation (SL+RSD) student MLP. We run all the experiments with ten seeds and report the mean accuracy.

Implementation Details: For the teacher NAGphormer [10], we have used the recommended settings given in the official Github repository. For all the MLP models, we have kept hidden dimensions as 64 and 2 using an Adam Optimizer with learning rate and weight decay for each dataset based on the official NAGphormer repository. We have run all experiments on a Tesla T4 GPU.

6 Results and Analysis

We conduct experiments on the downstream task of node classification. Table 1 compares the performance (%) of the proposed GraTeD-MLP with various KD methods for transductive while Table 2 showcases results in an inductive setting. We can conclude from the results that the proposed GraTeD-MLP outperforms the vanilla MLP by at least 20% in transductive settings. Specifically, the GraTeD-MLP achieves superior performance in almost all datasets in the transductive setting compared to the teacher model. The performance drop in the OGB-arxiv dataset is primarily due

Datasets	Teacher	Vanilla	Student		
	NAGphormer [10]	MLP	GLNN [12]	NOSMOG [13]	GraTeD-MLP
Cora [42]	89.65	63.61	81.55	84.05	**90.80**
PubMed [42]	89.71	69.65	87.33	86.94	**92.30**
Photo [43]	94.75	77.19	96.21	95.46	**98.21**
Computer [43]	91.22	66.87	92.90	92.73	**93.40**
Citeseer [42]	75.34	57.78	74.38	76.35	**80.11**
Aminer-CS [44]	52.95	32.34	46.15	41.28	**46.25**
OGB-Arxiv [45]	68.75	54.84	50.24	40.56	**58.17**

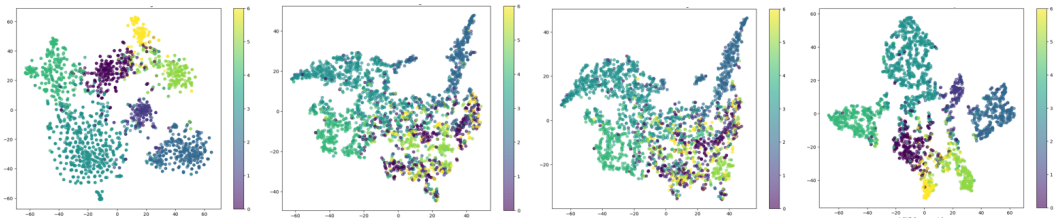
Table 1: Transductive results (%) for various KD methods, compared against the proposed GraTeD-MLP.

Datasets	Teacher	Vanilla	Student		
	NAGphormer [10]	MLP	GLNN [12]	NOSMOG [13]	GraTeD-MLP
Cora [42]	80.89	61.78	69.21	**71.10**	69.80
PubMed [42]	72.48	68.41	70.82	70.50	**71.48**
Photo [43]	90.00	78.55	89.45	89.90	**90.07**
Computer [43]	83.62	64.34	80.38	81.10	**81.74**
Citeseer [42]	68.43	59.50	68.26	71.07	**72.12**
Aminer-CS [44]	51.93	32.42	46.61	47.29	**48.14**
OGB-Arxiv [45]	68.57	54.84	50.24	51.33	**55.64**

Table 2: Inductive results (%) for various KD methods, compared against the proposed GraTeD-MLP.

to a distribution shift in training and test nodes [12]. The same is inherently more challenging for a graph-less model to capture. Yet, we still outperform compared to the other distilled MLP models. Though we outperform for Aminer-CS and OGB-Arxiv compared to other algorithms, overall performance on these datasets is the lowest. As recommended by Zhang et al. [12], a bigger MLP has been shown to perform better for large-scale datasets as they require larger models.

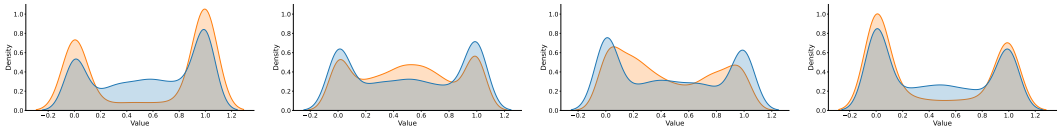
The proposed model demonstrates superior performance in inductive settings except for the Cora dataset. Overall, across distillation algorithms in the inductive setting, we observe a significant drop in performance in the student MLP from the teacher model (compared to the transductive counterpart). This is because the NAGphormer model is primarily developed for transductive settings. It focuses on global attention mechanisms, resulting in a lack of focus on local neighborhood information (pivotal for inductive settings). This results in a lack of local generalization, leading to lower inductive bias compared to models like GraphSage. Nevertheless, GraTeD-MLP still outperforms MLPs trained using other distillation methods. We leave this research direction as future work to be explored.



(a) NAGphormer [10] **(b)** GLNN [12] **(c)** NOSMOG [13] **(d)** GraTeD-MLP

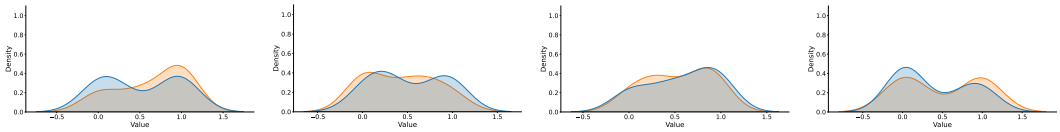
Figure 3: t-SNE plots for the embeddings generated from different algorithm.

How effective is the the proposed GraTeD-MLP in aligning the feature space: As shown by Figure 3(a-d), MLPs trained on existing distillation frameworks fail to adequately distinguish between different classes that the teacher model can (Figure 3a). On the contrary, Figure 3d illustrates that the proposed GraTeD-MLP framework effectively demonstrates strong separation between the classes, similar to the teacher NAGphormer. Furthermore, Figure 1 shows an increase in transferability by over 70% through the proposed GraTeD-MLP, compared to other models trained using the current hint-based methods. These two results effectively demonstrate the GraTeD-MLP's ability to align with the teacher's feature space.



(a) Hop 0 **(b)** Hop 1 **(c)** Hop 2 **(d)** Hop 3

Figure 4: KDE plots for teacher NAGphormer and reconstructed GraTeD-MLP attention maps for all 2468/2485 matching instances for Cora.



(a) Hop 0 **(b)** Hop 1 **(c)** Hop 2 **(d)** Hop 3

Figure 5: KDE plots for teacher NAGphormer and reconstructed GraTeD-MLP attention maps for all 17/2485 non matching instance for Cora.

For further analysis, we first reconstruct the attention matrix for each node by utilizing the U and V embeddings.Then, we plotted the Kernel Density Estimate (KDE) plot of average for all heads and attention dimension for each hop, specifying the importance of each hop. As observed in Figure 4(a-d), the peaks for both GraTeD-MLP and NAGphormer align, showing a positive correlation (and consequently, prediction matches). On the contrary, Figure 5(a-d) has a prediction mismatch, due to different attention for different hops.

How does the proposed GraTeD-MLP perform with respect to inference time: To demonstrate the efficiency of GraTeD-MLP, we analyze the tradeoff between prediction accuracy and model inference time on the Citeseer dataset in Figure 6. GraTeD-MLP can achieve high accuracy (80.11%) while maintaining a fast inference time. Specifically, compared to other models with similar inference times, GraTeD-MLP performs significantly better, while vanilla MLP and other baseline models can only achieve lower accuracy levels. This makes GraTeD-MLP 20×-40x faster (across databases), specifically 36x for Citeseer, than the teacher GT model. Additionally, GraTeD-MLP shows up to 22.33% improved performance over vanilla MLP (in Citeseer), demonstrating the effectiveness of our knowledge distillation approach. Further, as shown in Table 3, GraTeD-MLP outperforms with comparable inference time with different student models on a large-scale dataset of Ogbn-Products. Details of time complexity is presented in Appendix A.2.1.

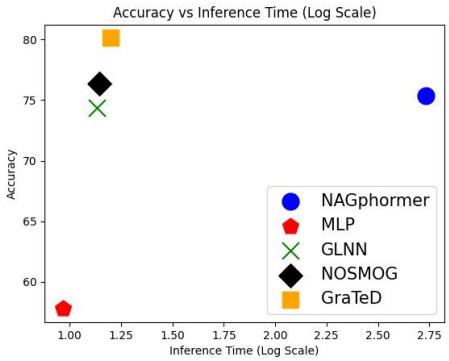


Figure 6: Accuracy (y axis)-time (x axis) trade-off.

7 Ablation Study

7.1 How consistent is the GraTeD-MLP predictions with graph topology

We leverage the approximate to the min-cut problem [12] to demonstrate how our GraTeD-MLP framework is superior to other distillation methods in encoding the graph structural information. The min-cut problem removes the minimum volume of edges by partioning N nodes in V into K disjoint

subsets. Accordingly, its expressed as [46].

$$max \frac{1}{K} \sum_{k=1}^{K=1} \frac{(C_k^T A C_k)}{(C_k^T D C_k)} \tag{9}$$

Bianchi et al. [47] demonstrated that replacing C with the model prediction output, $\hat{Y}$, would indicate consistency between model predictions and graph topology. A higher cut value indicates an increased capability of the model to capture the structural information. Hence, the cut value is as follows:

$$max \frac{tr(\hat{Y}^T A \hat{Y})}{(\hat{Y}^T D \hat{Y})} \tag{10}$$

Model	Accuracy (%)	Time/node (ms)
NAGphormer	77.54%	0.7500
MLP	44.81%	0.0089
GLNN	53.33%	0.0088
NOSMOG	53.35%	0.0090
Grated-MLP	**54.81**%	0.0160

Table 3: Comparison of model accuracy and time per node on a large scale dataset of Ogbn-Products.

Dataset	NAG -phormer	MLP	GraTeD -MLP
Cora [42]	0.864	0.740	**0.867**
Citeseer [42]	**0.95**	0.770	0.945
Pubmed [42]	0.899	0.862	**0.91**
A-comp [43]	**0.853**	0.675	0.851
A-photo [43]	**0.877**	0.673	0.86
Average	0.8886	0.744	0.8866

Table 4: Cut value for NAGphormer, MLP and the proposed GraTeD-MLP on five datasets in the transductive setting.

Based on the above, Table 4 shows that GraTeD-MLP can better capture structural information than a vanilla MLP. On average, it is as good as the teacher NAGphormer model.

7.2 How does GraTeD-MLP perform with respect to different model components

As our GraTeD-MLP model contains four essential components, we conduct ablation studies to analyze the importance of each component by removing them independently.

Dataset	w/o CE	w/o SL	w/o U	w/o V	GraTeD	Δ_{CE}	Δ_{SL}	Δ_U	Δ_V
Cora	90.54%	76.68%	90.16%	90.37%	**90.80%**	↑0.36%	↑14.12%	↑0.64%	↑0.43%
Citeseer	78.87%	48.43%	76.61%	77.50%	**80.11%**	↑2.24%	↑31.68%	↑3.50%	↑2.61%

Table 5: Ablation study: Accuracy of distilled model under different settings.

As we can observe from Table 5, the SL loss contributes the most to performance because it is the most pivotal loss for knowledge distillation frameworks. The structural components U and V contribute significantly to performance, showing the importance of distilling structure information. We also notice that the U component contributes more to the performance than V because U encodes hop encodings while V primarily encodes directions. We also observe that even without the CE loss, we can still achieve decent performance, demonstrating the effectiveness of U and V losses. Finally, GraTeD-MLP achieves the best performance, demonstrating the effectiveness of the model.

8 Conclusion

In this paper, we aim to address the challenge of cross-architecture distilling by bridging the gap between graph-based transformer architecture and MLPs. We propose GraTeD-MLP, which assists in developing deployable structure-aware graph-independent models by encoding attention. GraTeD-MLP explicitly addressed the lack of transferability and leveraged the low-rank nature of the attention matrix for the same. We showcase extensive experimentation over various small-scale and large-scale datasets that demonstrate the effectiveness of GraTeD-MLP. Using GraTeD-MLP, we observed an increase in accuracy of up to 25%. Furthermore, t-SNE and CKA plots showcase better-learned embedding/class separation and transferability, respectively. **From an ethical standpoint**, while knowledge distillation aids model deployment of complex high-performing models, it may sometimes perform lower than the teacher model. Hence, we recommend exercising caution and responsibility to ensure positive and socially beneficial outcomes of the lightweight distilled GraTeD-MLP.

References

[1] Yanfu Zhang, Shangqian Gao, Jian Pei, and Heng Huang. Improving social network embedding via new second-order continuous graph neural networks. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 2515–2523, 2022. 1

[2] Kanchan Jha, Sriparna Saha, and Hiteshi Singh. Prediction of protein–protein interaction using graph neural networks. *Scientific Reports*, 12(1):8360, 2022. 1

[3] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2022. 1

[4] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017. 1

[5] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020. 1, 3

[6] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in neural information processing systems*, 34:28877–28888, 2021.

[7] Dexiong Chen, Leslie O'Bray, and Karsten Borgwardt. Structure-aware transformer for graph representation learning. In *International Conference on Machine Learning*, pages 3469–3489. PMLR, 2022. 1

[8] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018. 1

[9] Hoang Nt and Takanori Maehara. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*, 2019. 1

[10] Jinsong Chen, Kaiyuan Gao, Gaichao Li, and Kun He. Nagphormer: A tokenized graph transformer for node classification in large graphs. In *The Eleventh International Conference on Learning Representations*, 2022. 1, 3, 4, 6, 7, 13, 14

[11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2, 3, 5

[12] Shichang Zhang, Yozen Liu, Yizhou Sun, and Neil Shah. Graph-less neural networks: Teaching old mlps new tricks via distillation. In *International Conference on Learning Representations*, 2021. 2, 3, 4, 6, 7, 8

[13] Yijun Tian, Chuxu Zhang, Zhichun Guo, Xiangliang Zhang, and Nitesh Chawla. Nosmog: Learning noise-robust and structure-aware mlps on graphs. In *NeurIPS 2022 Workshop: New Frontiers in Graph Learning*, 2022. 2, 3, 4, 6, 7

[14] Zhiwei Hao, Jianyuan Guo, Kai Han, Yehui Tang, Han Hu, Yunhe Wang, and Chang Xu. One-for-all: Bridge the gap between heterogeneous architectures in knowledge distillation. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 3

[15] Adrian Kochsiek and Rainer Gemulla. Parallel training of knowledge graph embedding models: a comparison of techniques. *Proceedings of the Very Large Data Base Endowment*, 15(3): 633–645, 2021. 3

[16] Yuhe Bai. A semantic partitioning method for large-scale training of knowledge graph embeddings. In *ACM Web Conference*, pages 573–577, 2023.

[17] Sudipta Modak, Aakarsh Malhotra, Sarthak Malik, Anil Surisetty, and Esam Abdel-Raheem. Cpa-wac: Constellation partitioning-based scalable weighted aggregation composition for knowledge graph embedding. pages 3504–3512, 2024.

[18] Jay Puja, Hui Miao, Lise Getoor, and William W Cohen. Ontology-aware partitioning for knowledge graph identification. In *Automated knowledge base construction*, pages 19–24, 2013. 3

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 3

[20] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[21] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019. 3

[22] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021. 3

[23] Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515, 2022. 3

[24] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets, 2015. 3

[25] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1921–1930, 2019.

[26] Zehao Huang and Naiyan Wang. Like what you like: Knowledge distill via neuron selectivity transfer. *arXiv preprint arXiv:1707.01219*, 2017.

[27] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4133–4141, 2017.

[28] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788, 2020.

[29] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[30] Gustavo Aguilar, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. Knowledge distillation from internal representations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7350–7357, 2020. 3

[31] Yufan Liu, Jiajiong Cao, Bing Li, Weiming Hu, Jingting Ding, and Liang Li. Cross-architecture knowledge distillation. In *Proceedings of the Asian conference on computer vision*, pages 3396–3411, 2022. 3

[32] Yiding Yang, Jiayan Qiu, Mingli Song, Dacheng Tao, and Xinchao Wang. Distilling knowledge from graph convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7074–7083, 2020. 3

[33] Cheng Yang, Jiawei Liu, and Chuan Shi. Extract the knowledge of graph neural networks and go beyond it: An effective knowledge distillation framework. In *Proceedings of the web conference 2021*, pages 1227–1237, 2021. 6

[34] Bencheng Yan, Chaokun Wang, Gaoyang Guo, and Yunkai Lou. Tinygnn: Learning efficient graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1848–1856, 2020.

[35] Zhichun Guo, Chunhui Zhang, Yujie Fan, Yijun Tian, Chuxu Zhang, and Nitesh V Chawla. Boosting graph neural networks via adaptive knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 7793–7801, 2023.

[36] Seunghyun Lee and Byung Cheol Song. Graph-based knowledge distillation by multi-head attention network. In *30th British Machine Vision Conference, BMVC 2019*, 2020. 3

[37] Wenqing Zheng, Edward W Huang, Nikhil Rao, Sumeet Katariya, Zhangyang Wang, and Karthik Subbian. Cold brew: Distilling graph node representations with incomplete or missing neighborhoods. In *International Conference on Learning Representations*, 2021. 3, 4

[38] Yong-Min Shin and Won-Yong Shin. Propagate & distill: Towards effective graph learners using propagation-embracing mlps. In *The Second Learning on Graphs Conference*, 2023.

[39] Lirong Wu, Haitao Lin, Yufei Huang, Tianyu Fan, and Stan Z Li. Extracting low-/high-frequency knowledge from graph neural networks and injecting it into mlps: an effective gnn-to-mlp distillation framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 10351–10360, 2023. 3, 4

[40] Lirong Wu, Haitao Lin, Yufei Huang, and Stan Z Li. Quantifying the knowledge in gnns for reliable distillation into mlps. In *International Conference on Machine Learning*, pages 37571–37581. PMLR, 2023. 3

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 4

[42] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008. 6, 7, 9

[43] Galileo Namata, Ben London, Lise Getoor, Bert Huang, and U Edu. Query-driven active surveying for collective classification. In *10th international workshop on mining and learning with graphs*, volume 8, page 1, 2012. 6, 7, 9

[44] Wenzheng Feng, Yuxiao Dong, Tinglin Huang, Ziqi Yin, Xu Cheng, Evgeny Kharlamov, and Jie Tang. Grand+: Scalable graph random neural networks. In *Proceedings of the ACM Web Conference 2022*, pages 3248–3258, 2022. 6, 7

[45] 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. *Advances in neural information processing systems*, 33:22118–22133, 2020. 6, 7

[46] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means: spectral clustering and normalized cuts. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 551–556, 2004. 9

[47] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling. In *International conference on machine learning*, pages 874–883. PMLR, 2020. 9

[48] Gilbert Strang. *Linear algebra and its applications*. Thomson, Brooks/Cole, Belmont, CA, 2006. ISBN 0030105676 9780030105678 0534422004 9780534422004. URL http://www.amazon.com/Linear-Algebra-Its-Applications-Edition/dp/0030105676. 13

[49] Junhan Yang, Zheng Liu, Shitao Xiao, Chaozhuo Li, Defu Lian, Sanjay Agrawal, Amit Singh, Guangzhong Sun, and Xing Xie. Graphformers: Gnn-nested transformers for representation learning on textual graph. *Advances in Neural Information Processing Systems*, 34:28798–28810, 2021. 15

[50] Jiawei Zhang, Haopeng Zhang, Congying Xia, and Li Sun. Graph-bert: Only attention is needed for learning graph representations. *arXiv preprint arXiv:2001.05140*, 2020. 15

A Appendix

A.1 CKA

Let $H \in R^{nxd_1}$ and $\hat{H} \in R^{nxd_2}$ correspond to the hidden dimension embedding for two models whose dimensions are d_1 and d_2 respectively. The CKA similarity is defined as

$$CKA(K, L) = \frac{HSIC(\hat{K}, \hat{L})}{\sqrt{HSIC(\hat{K}, \hat{K})}\sqrt{HSIC(\hat{L}, \hat{L})}} \tag{11}$$

where $\hat{K}$ and $\hat{L}$ correspond to the centered Kernel matrices which is obtained by multipliying both K and L with a centering matrix $C_n = I_n - \frac{1}{n}11^T$ where $K = HH^T$ and $L = \hat{H}\hat{H}^T$ in the case of a linear kernel and $K = exp(-\frac{\|H_i - H_j\|}{2\sigma^2})$ and $L = exp(-\frac{\|H_i|\hat{-}\hat{H}_j[\|}{2\sigma^2})$ in the case of RBF kernel,

Algorithm 1 GraTeD-MLP Training

1: **Input:** Graph $G = (V, E)$, node features X, labels Y, pre-trained GT teacher model T
2: **Output:** Trained GraTeD-MLP student model S
3: Initialize multi-branch MLP student model S
4: // Phase 1: Attention Matrix Decomposition
5: **for** each attention head i in teacher model **do**
6: $\quad A^i \leftarrow \text{GetAttentionMatrix}(T, G)$
7: $\quad U^i, \Sigma^i, V^i \leftarrow \text{SVD}(A^i)$
8: **end for**
9: // Phase 2: Multi-branch Training
10: **while** not converged **do**
11: $\quad \hat{U\Sigma} \leftarrow \text{StructureStudent1}(X)$
12: $\quad \hat{V} \leftarrow \text{StructureStudent2}(X)$
13: $\quad \hat{E} \leftarrow \text{FeatureStudent}(X)$
14: $\quad \hat{Y} \leftarrow \text{GatingMechanism}(\hat{U\Sigma}, \hat{V}, \hat{E})$
15: $\quad$ Compute losses: $\mathcal{L}_{CE}$, $\mathcal{L}_{SL}$, $\mathcal{L}_{U\Sigma}$, $\mathcal{L}_V$
16: $\quad$ Update S using $\mathcal{L} = \lambda_1\mathcal{L}_{CE} + \lambda_2\mathcal{L}_{SL} + \lambda_3\mathcal{L}_{U\Sigma} + \lambda_4\mathcal{L}_V$
17: **end while**
18: **return** S

where HSIC is the Hilbert-Schmidt interdependence criterion and measures the dependence of two sets of variables using kernel matrices and is calculated as

$$HSIC(K, L) = \frac{trace(\hat{K}\hat{L})}{(n-1)^2} \tag{12}$$

A.2 Theoretical Guarantees and Training Procedure

Algorithm 1 presents the complete training procedure for GraTeD-MLP. Briefly, the algorithm consists of two main phases: (1) attention matrix decomposition from the teacher GT model, and (2) multi-branch MLP training with the decomposed components.

The attention matrix present in the Nagphormer is of the form:

$$A = softmax\left(\frac{QK^T}{\sqrt{d_K}}\right)\mathcal{V} \tag{13}$$

Here, for Napghormer according to Chen et al. [10], the attention matrix is computed for each node. The sequence comprises of the node's features and the embeddings from the hop2token algorithm, and would be of the size $(K + 1) \times d$. Here K is the number of hops, and d is the hidden size.

First, we obtain the attention matrices from each head of the pre-trained GT teacher model (lines 1-3 of the Algorithm). For each attention matrix, we perform SVD to decompose it into U, Σ, and V components (line 4). According to [48], the columns of U and V represent orthonormal basis of column and row space of A respectively. These components capture the structural information encoded in the attention mechanism. In this context, the U matrix serves as a singular basis for the feature space of nodes and their hop embeddings derived from the Hop2 token algorithm. Each column in U represents a distinct attention pattern across the input node-hop structure, encapsulating key "directions" in the attention feature space. Meanwhile, each row in V specifies a weight vector for the attention dimensions, illustrating how much each direction in U influences the original attention matrix. Thus, V encodes the relationships between hops, capturing the structural dependencies and interactions among them across the graph.

The multi-branch MLP architecture is then trained using these decomposed components. The first two branches (Structure Students) are trained to predict the $U\Sigma$ and V components, respectively (lines 7-8), while the third branch (Feature Student) learns to map the input node features to the desired embedding space (line 9). The final prediction is obtained by combining the outputs of all three branches (line 10). The model is optimized using four different loss terms:

$\mathcal{L}_{CE}$: Cross-entropy loss between predicted labels and ground truth for labeled nodes.
$\mathcal{L}_{SL}$: Soft label loss measuring KL divergence between student and teacher predictions.

$\mathcal{L}_{U\Sigma}$: L_2 loss between predicted and teacher's $U\Sigma$ components.
$\mathcal{L}_V$: L_2 loss between predicted and teacher's V components.

These losses are weighted and combined to form the final optimization objective (lines 11-12). The algorithm terminates when convergence is reached, returning the trained GraTeD-MLP model that efficiently captures both structural and feature-level information from the teacher GT model while maintaining significantly lower computational complexity.

A.2.1 Time complexity analysis

According to [10], the time complexity of the teacher Nagphormer is dominated by the self attention module, which corresponds to $O(N(K+1)^2d)$. Here, N denotes the number of nodes, K denotes the number of hops, and d represents parameter size dimension.

For the proposed GraTeD-MLP, the first layer transforms the d-dimensional input into an m-dimensional feature space with complexity $O(md)$. This then branches out into three paths, each with dimensions H, contributing a complexity of $O(3mH)$. After concatenation, the output is mapped to p outputs with complexity $O(3Hp)$. Summing up for N nodes, the total time complexity is $O(N(md + 3mH + 3Hp))$. As H and p are generally lesser than the input dimension, hence, it can be reduced to $O(Nmd)$.

For a Nagphormer [10], the above time complexity applies in a transductive setting where all nodes are visible during training. However, in real-world applications, where inferencing is performed on never-before-seen nodes, we would need to compute the attention matrix using the computationally expensive Hop2Token algorithm. This algorithm has a time complexity of $O(N^2Kd)$, which is quadratic and makes Nagphormer less viable for large-scale applications. In contrast, our algorithm maintains the same time complexity during inference, regardless of whether it is a transductive or an inductive setting. This is because it only requires node features as input, while the structural insights from Nagphormer are already integrated during training.

A.3 Detailed Accuracy-Time-Memory Evaluation

We conducted the experiments with more number of parameters. In this setting, we increased parameters in GLNN and NOSMOG to same as we have in Grated-MLP. As we observe in the Figure 7, even with similar amount of parameters, Grated-MLP is able to outperform NOSMOG-512 and GLNN-512.

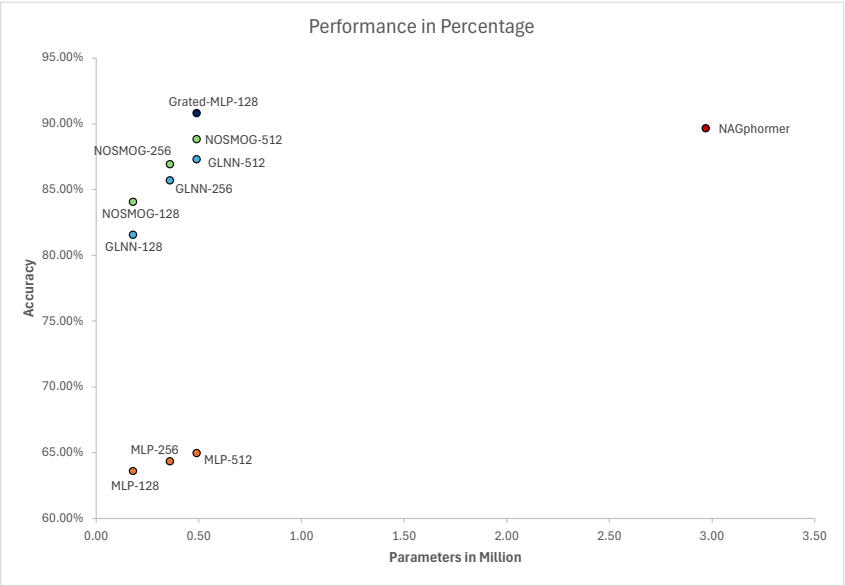


Figure 7: Model parameters vs performance

Further, as indicated in the Table 6, the GraTeD-MLP contains, on average, 3 to 4 times fewer parameters than the teacher model. Despite fewer parameters, it consistently outperforms the teacher model, underscoring its robustness and effectiveness.

Dataset	NAGphormer Param	NAGphormer Acc	Grated-MLP Param	Grated-MLP Acc
Cora	2,973,705	89.65%	497,287	**90.80**%
Citeseer	8,341,256	75.34%	2,358,022	**80.11**%
Pubmed	2,494,981	89.70%	689,667	**92.30**%

Table 6: Comparison of NAGphormer and Grated-MLP parameters and accuracy across datasets.

A.4 Additional Experiments

To assess the robustness of the proposed paradigm across various graph transformer (GT) architectures, we conducted additional experiments using two distinct types of GTs: a hybrid GT [49] and a pure GT. For the pure GTs, which rely solely on attention mechanisms, we adjusted the settings in Graph-BERT [50] to ensure that the model exclusively utilized attention. Furthermore, it is important to note that NAGphormer is also classified as a hybrid GT, as it employs attention on hop embeddings, akin to message passing.

As observed in Table Table 7, GraTeD-MLP can mimic both types of GTs as GraTeD-MLP is a student based on only MLP layers so it can't be differentiated between attention with or without message passing. In a pure GT context, the proposed GraTeD-MLP outperforms by 4.53% for the Citeseer and 0.30% the Cora dataset, respectively. Hence, this proves that the GraTeD-MLP can mimick the attention matrix irrespective of where attention is coming from.

Dataset	NAGphormer	Grated-MLP (NAGphormer)	Hybrid GT	Grated-MLP (Hybrid GT)	Pure GT	Grated-MLP (Pure GT)
Cora	89.65%	90.80%	88.72%	88.85%	82.02%	82.32%
Citeseer	75.34%	80.11%	70.98%	76.11%	64.36%	68.89%

Table 7: Comparison of Grated-MLP performance for different types of GTs