Pruned teachers are friendly teachers: improving knowledge distillation in GAT networks

Antonia Boca
Department of Computer Science
University of Cambridge
aib36@cam.ac.uk

Abstract

Knowledge distillation (KD) is a popular technique used to compress large machine learning models into smaller ones that can be more easily deployed on devices with limited resources. A recent line of work investigates "student-friendly" teachers that are more appropriate for knowledge transfer; pruned networks constitute one such example, with recent results suggesting that students trained with pruned models outperform models trained with unpruned teachers. While this line of work looks promising for future deployment of performant networks on resource-limited environments, little work has been done on investigating whether knowledge distillation using pruned networks can be used for Graph Neural Networks (GNNs). This project investigates whether pruned GNNs are better teachers than their unpruned counterparts. Results indicate that pruning GAT networks improves the distillation process on transductive node classficaition tasks. This is promising for many memory-constrained applications that require GNNs to perform inference, such as mobile navigation services and privacy-driven local recommender systems on apps.

The code can be found at: `https://github.com/semiluna/l46_project`.

1 Introduction

Graph neural networks (GNNs) are a type of neural network that can operate on data represented in the form of graphs to perform tasks such as node and graph classification, and link prediction. All GNNs use a form of *neural message passing*, in which vector messages are exchanged between nodes and updated using neural networks [Gilmer et al., 2017]. Due to the structure-aware exploitation of graphs, GNNs are hard to parallelise and are prohibitively expensive to train and deploy on large graphs. However, standard model compression techniques, such as knowledge distillation [Hinton et al., 2015] and pruning [Wang et al., 2020] are not straightforward to apply to GNNs. For example, some classification tasks on GNNs involve training on a single graph; when this is the case, one may ask whether the original graph could be part of the pruning pipeline as well, or whether node embeddings may also be compressed for further speed-ups. While there has been recent work in compressing such models [Chen et al., 2021, Yang et al., 2020, Joshi et al., 2022], a unified framework has not been proposed, with different methods performing better on some graphs and worse on others.

A different line of recent work has been investigating "student-friendly" teachers for the task of knowledge distillation [Mirzadeh et al., 2020, Park et al., 2021]. One interesting proposal comes from Park and No [2022], who argue that pruned networks are better teachers than their unpruned counterparts because they act as regularisers during distillation. However, their experiments are limited to pruning and distlling convolutional neural networks. As such, the aim of this project is to extend the hypothesis presented in Park and No [2022] by investigating whether pruned GNNs are

Preprint. Under review.

better teachers for knowledge distillation, and ultimately advancing the work on a unified compression pipeline for GNNs.

1.1 Contributions

The contributions of this project are the following:

1. I present and discuss various methods for pruning and distilling GNNs.
2. I implement the **prune-then-distill** pipeline introduced by Park and No [2022] for the task of transductive node classification using GAT networks [Veličković et al., 2017].
3. I run experiments on pruned-then-distilled models and show that these perform better than unpruned, distilled models, thereby finding "student-friendly" teachers for GAT networks.

2 Theoretical background

2.1 Graph neural networks

Graph neural networks are characterised by the *neural message passing* framework, in which vector messages are exchanged between nodes and then updated using neural networks. At message passing iteration k, a hidden embedding $\mathbf{h}_u^{(k)}$ corresponds to each node $u \in \mathcal{V}$, and an update happens according to information aggregated from u's graph neighbourhood $\mathcal{N}(u)$, as seen in Equation 2.

$$\mathbf{h}_u^{(k+1)} = \text{UPDATE}^{(k)}(\mathbf{h}_u^{(k+1)}, \text{AGGREGATE}^{(k)}(\{\mathbf{h}_v^{(k+1)}, \forall v \in \mathcal{N}(u)\})) \tag{1}$$

$$= \text{UPDATE}^{(k)}(\mathbf{h}_u^{(k+1)}, \mathbf{m}_{\mathcal{N}(u)}^{(k)}) \tag{2}$$

Note that in general the AGGREGATE function must be permutation-invariant.

Graph Attention Networks. One popular kind of GNNs, introduced by Veličković et al. [2017], are Graph Attention Networks (GATs). For these networks, the UPDATE and AGGREGATE functions form the following attentional operator:

$$\mathbf{h}_u^{(k+1)} = \alpha_{u,u}\Theta\mathbf{h}_u^{(k)} + \sum_{v \in \mathcal{N}(u)} \alpha_{u,v}\Theta\mathbf{h}_v^{(k)}, \tag{3}$$

where $\alpha_{u,v}$ is an attention coefficient computed as

$$\alpha_{u,v} = \frac{\exp(\text{LeakyReLU}(\mathbf{a}^T[\Theta\mathbf{x}_u || \Theta\mathbf{x}_v]))}{\sum_{k \in \mathcal{N}(u)\cup\{u\}} \exp(\text{LeakyReLU}(\mathbf{a}^T[\Theta\mathbf{x}_u || \mathbf{x}_k]))}. \tag{4}$$

Node classification. One important task in GNNs is that of node classification, where the goal is to predict class labels for some nodes in a graph. For example, in a social network graph, the nodes might represent individuals, and the task could be to predict the occupation of each individual based on their attributes and connections in the graph. Node classification can be further categorised as either *transductive* or *inductive*. In the former setting, the training, validation and test nodes are all on the same graph and the entire graph is made available to the model during training, but the loss is computed only using the training nodes. In the latter setting, some graphs are completely unseen during training.

2.2 Pruning

Pruning is a method for model compression that removes weights from a trained model [LeCun et al., 1989, Hassibi and Stork, 1992, Han et al., 2015, Li et al., 2016] without harming accuracy. Pruned models are smaller, therefore reducing energy consumption and inference time, making them more amenable to deployment on devices with limited resources. Additionally, Frankle and Carbin [2018] show that neural networks contain *winning lottery tickets*: subnetworks that, when trained in isolation, can reach a similar performance to the original network, due to "lucky" initial weights that make training effective. It turns out that these subnetworks can be identified using the following iterative pruning approach, described in Frankle and Carbin [2018]:

1. Randomly initialise a neural network $f(x; \theta_0)$.
2. Train the network for j iterations to get parameters θ_j.
3. Prune $p\%$ of the network's parameters.
4. Reset the remaining parameters to their original θ_0 values. Return to step 2.

Pruning GNNs. Chen et al. [2021] generalised the lottery ticket hypothesis to GNNs; one of the novel ideas of their approach is to prune both the *graph* and the *model* associated with it at the same time, showing that *graph lottery tickets* (GLTs) exist for transductive node classification and link prediction tasks. More recently, Liu et al. [2022] proposed an alternative to the LTH pipeline for pruning GNNs, that requires fewer computational resources and can prune node embeddings.

In this project, I will use the pruning pipeline described by Chen et al. [2021] for node classification on the Cora, Citeseer and Pubmed datasets [Kipf and Welling, 2016a] using GATs.

2.3 Knowledge distillation

The goal of knowledge distillation is to transfer the knowledge from the large model, or "teacher," to the smaller model, or "student," so that the student model can perform nearly as well as the teacher on the task for which it was trained. Knowledge distillation allows the superior performance of large models to be transferred to smaller networks that can be deployed in resource-constrained environments. KD is usually implemented by modifying the objective of the student in a way that takes into account the predictions of the teacher.

logit-based KD. For each node $i \in \mathcal{V}$, the original KD objective [Hinton et al., 2015] uses the cross-entropy loss or KL-divergence to match the output logits of a student z_i^S to those of a teacher z_i^T, scaled by a temperature γ_1 that softens the logits of the teacher:[1]

$$\mathcal{L}_{KD} = \sum_{i \in \mathcal{V}} \mathcal{H}(\text{softmax}(\frac{z_i^T}{\gamma_1}), \text{softmax}(\frac{z_i^S}{\gamma_1})) \tag{5}$$

Local Structure Preserving Distillation. LSP [Yang et al., 2020] is a GNN distillation objective that trains the student model to preserve the local structure of graph data from the teacher's node embeddings space. The objective can be formalised as follows:

$$\mathcal{L}_{\text{LSP}} = \sum_{i \in \mathcal{V}} \mathcal{D}_{\text{KL}}(\text{softmax}_{(i,j) \in \mathcal{E}}(\mathcal{K}(f_i^S, f_j^S)) \;||\; \text{softmax}_{(i,j) \in \mathcal{E}}(\mathcal{K}(f_i^T, f_j^T))). \tag{6}$$

Intuitively, the local structure is defined as the set of parameterised pairwise distances between neighbouring nodes. The student then tries to match the teacher's distribution of distances. The distance between a pair of nodes is calculated via a kernel function $\mathcal{K}$ that can be the Euclidean distance between the two node features. Yang et al. [2020] find, however, that the RBF kernel performs best in practice:

$$\mathcal{K}(f_i, f_j) = \exp -\frac{1}{2\sigma} ||f_i - f_j||^2 \tag{7}$$

Other distillation objectives. Other distillation objectives for GNNs focus on latent representations of the global structure to the student model as presented by Joshi et al. [2022]. Their method, Graph Constrastive Representation Distillation, generalises Contrastive Representation Distillation [Tian et al., 2019] to GNNs. The authors also adapt two other mimicking techniques for convolutional neural networks: FitNet [Romero et al., 2014] and Attention Transfer [Zagoruyko and Komodakis, 2016].

In this project, I will use **logit-based KD** and **LSP** as the distillation objectives when training student GAT networks.

[1] Formulation adapted from Joshi et al. [2022]

3 Implementation

This project implements and analysises the following **prune-then-distll** pipeline:

1. Train a teacher GAT network [Veličković et al., 2017] to perform *transductive node classification* on the Cora, Citeseer, and Pubmed datasets.
2. Prune this teacher according to Chen et al. [2021].
3. Train a student GAT network with and without distillation. The distillation objectives are logit-based KD [Hinton et al., 2015] and LSP [Yang et al., 2020]. The teachers are represented by large models pruned up to 20 different sparsity levels.
4. Compare student networks trained with different strategies.

3.1 Pruning the teacher

For the first and second stages of the project I adapt the codebase[2] of Chen et al. [2021]. A trained teacher model will be pruned according to the **unified GNN sparsification** framework, which consists of three steps: training, pruning and rewinding the weights. I will briefly summarise these steps below.

(Training) Given a graph $\mathcal{G}$ with adjacency matrix $\mathbf{A}$ and network weights Θ, the authors introduce two differentiable masks m_g and m_θ that indicate the connections that should be pruned. Mask m_g indicates how the input graph should be pruned, while m_θ indicates how the model should be pruned. Then, at a training step, the following objective needs to be optimised:

$$\mathcal{L}_{UGS} = \mathcal{L}(\{m_g \odot \mathbf{A}, \mathbf{X}\}, m_\theta \odot \Theta) + \gamma_1 ||m_g||_1 + \gamma_2 ||m_\theta||_1, \tag{8}$$

where γ_1 and γ_2 are hyperparameters for l_1 sparsity regularisation.

(Pruning) When training is finished, the lowest values in both masks are set to zero (20% of m_θ and 5% of m_g).

(Rewinding) The remaining weights are rewound to their initial values and the model is retrained with the remaining connections, as per the original iterative pruning algorithm presented in Frankle and Carbin [2018].

Saving pruned teachers. Each of the iterations presented above cuts 20% of the remaining model weights and 5% of the remaining graph edges. I perform the three-step iteration above 20 times, reaching a graph with only 35.85% of the original edges and a model with only 1.15% of the original weights. I save the pruned teacher associated with each iteration, in order for them to be used in the distillation stage.

3.2 Training the student

The distillation pipeline is adapted from the codebase[3] of Yang et al. [2020]. I made the following changes to their codebase in order to fit it into my pipeline:

- I replaced their implementation of the GAT network [Veličković et al., 2017] with the implementation in the paper of Chen et al. [2021]. This way, the latent features of the teacher (as used in Equation 6) can be properly compared to the student's features.
- I adapted their distillation pipeline to support *transductive* node classification tasks on the Cora, Citeseer and Pubmed datasets, since the original paper was concerned with inductive node classification on the protein-protein iteraction dataset [Zitnik and Leskovec, 2017].
- I implemented logit-based KD (as presented in Equation 5) as a baseline distillation technique. I am interested in the potential benefits of distilling latent representations from teachers to students, especially when the teacher is pruned; to this end, I will compare LSP distillation to logit-based KD.

[2] `https://github.com/VITA-Group/Unified-LTH-GNN/`

[3] `https://github.com/ihollywhy/DistillGCN.PyTorch`

3.3 Summary of architectures and datasets

The teacher architecture was chosen following Chen et al. [2021]. GAT Networks have been shown to perform best on Cora, Citeseer and Pubmed when they have *2 layers* and *8 attention heads* per layer, according to Veličković et al. [2017]. The student architecture was chosen following Yang et al. [2020]. Although it has significantly fewer parameters (see Tables 1 and 3), it has double the depth of the teacher network. I hypothesise that this depth gives the student network enough capacity to reach on-par accuracy with the teacher when distilled properly, as I show in Section 4.2.

Table 1: Summary of teacher and student architectures.

Model	Layers	Attention heads	Hidden features
Teacher	2	8,8	512,512
Student	4	2,2,2,2	68,68,68,68

Datasets. Cora (first introduced by McCallum et al. [2000]), Citeseer (first introduced by Giles et al. [1998]) and Pubmed (first introduced by Sen et al. [2008]) are citation networks commonly used for evaluating the performance of GNNs. Table 2 summarises some graph statistics. All three datasets contain *a single graph*, making these datasets standard benchmarks for transductive node classification and link prediction tasks.

Table 2: Graph dataset statistics.

Dataset	Nodes	Edges	Avg. Degree	Features	Classes
Cora	2,708	5,429	3.88	1,433	7
Citeseer	3,327	4,732	2.84	3,703	6
Pubmed	19,717	44,338	4.50	500	3

Size of models. Since GNNs are structure-aware, the size of a model depends on the dataset it was trained on. Table 3 summarises the parameter count of the **unpruned** teacher and student networks.

Table 3: Summary of the number of parameters on each of the three datasets.

Model	Trainable parameters		
	Cora	Citeseer	Pubmed
Teacher	5.9M	15M	2M
Student	0.2M	0.5M	0.1M

4 Experiments

All models were trained on my personal laptop (Apple M2 chip, 24 GB). The main goal of this project is to analyse the effect pruning has on the distillation ability of teachers. To achieve this goal, I first train and prune GAT models up to 20 different sparsity levels on the Cora, Citeseer and Pubmed datasets. I then distill the knowledge of all these models into smaller GAT networks and compare the effect pruning had on the distillation process.

4.1 Teacher results

The training hyperparameters for pruning the teachers follow Chen et al. [2021]; I used the Adam optimiser [Kingma and Ba, 2014] with a learning rate of 0.01 and a weight decay of $5e^{-4}$. I used a dropout of 0.6 within each GAT layer. Figure 1 summarises the test accuracies of the pruned teachers for each dataset. We can notice that the sparsified networks perform well, with almost non-existent performance drops. Interestingly, for the Pubmed dataset, the performance increases at every level of

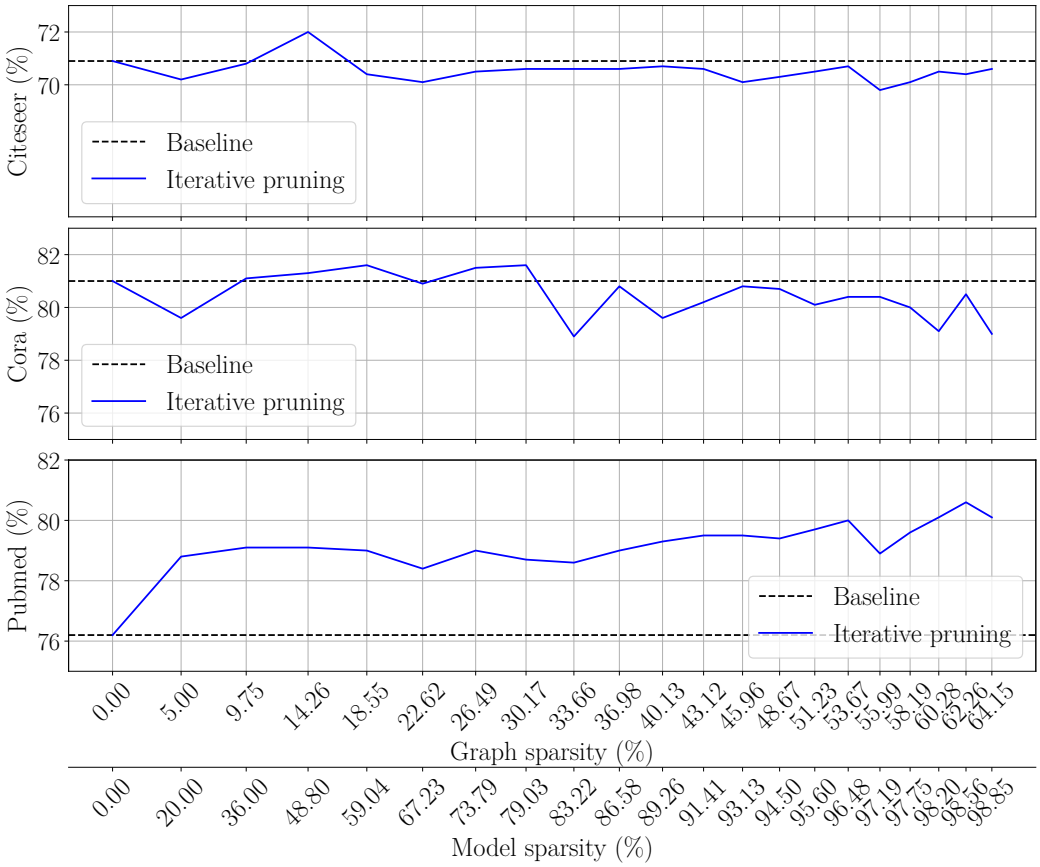


Figure 1: Test accuracy of pruned teacher models on the three datasets. The iterative pruning algorithm is adapted from Chen et al. [2021] and presented in Section 3.1; at each iteration I train a model on its associated graph, prune 20% of the model mask's weights and 5% of the graph mask's weights, rewind the remaining model weights to their starting values and re-train the sparsified network. Baseline represents the test accuracy of an unpruned model.

sparsification. This may be explained by the fact that Pubmed is the biggest citation network of the three, containing many redundant citation edges between massively influential papers.

Graph Lottery Tickets. The results I obtain are similar to those in Chen et al. [2021]. We can identify the presence of graph lottery tickets even at **62.26%** graph sparsity across all datasets.

4.2 Student results

All student models are trained with Adam [Kingma and Ba, 2014] and a learning rate of 0.005. When performing distillation using logit-based KD (section 2.3), I set the temperature γ_1 to 10 and weigh the teacher loss and the student loss equally. For LSP, I use the RBF Kernel (as recommended by Yang et al. [2020]) with $\sigma = 5$. Figure 2 plots the various test accuracies of distilled models w.r.t. to the sparsity of their teacher. I distill every teacher 3 times and report the resulting models' test accuracy mean and standard deviation. The student baseline is represented by a model trained without distillation. The teacher baseline is the same as in Figure 1. The rest of this section discusses the key takeaways from distilling pruned models.

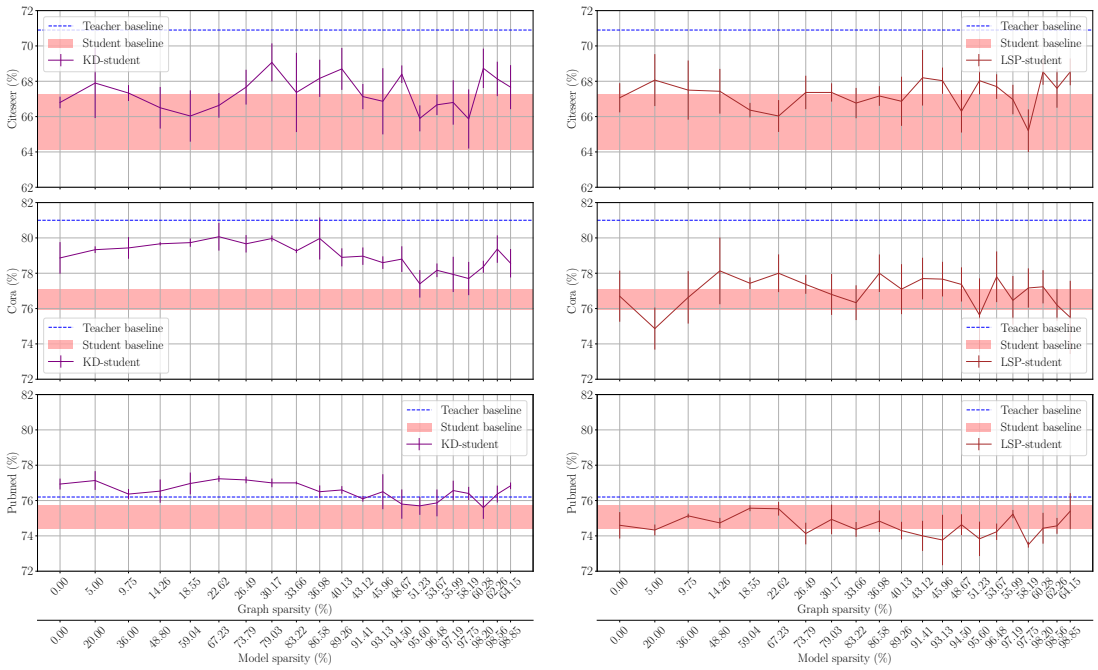
Logit-based KD improves model accuracies. One of the trends we can notice in Figure 2a is that logit-based KD improves model performance w.r.t to the student baseline for *all* teacher sparsity levels. The biggest impact can be noticed for the Cora dataset, where one distilled model has an improvement of more than 3% over the baseline. Distillation using the LSP loss does not have an overall trend, but it consistently makes models perform worse for the Pubmed dataset, even though

all pruned Pubmed teachers are better than their unpruned conterpart. I discuss why this may be the case further down.

Pruned teachers are generally better teachers. Perhaps most importantly, when using the logit-based KD loss, pruned teachers distill information better than their unpruned counterparts. Figure 2 shows that the performance of pruned-then-distilled models increases compared to the unpruned distilled model for a wide range of sparsity levels across all three datasets. In particular, Table 4 presents a selection of students and their test accuracies for different teacher sparsity lavels. It is worth noting that the teacher model still contains the two sparsity masks, m_g and m_θ. This means that for every graph input, while the student sees the entire graph, the teacher still "masks" some edges before inference. I believe that this particular behaviour helps the student, since this way the teacher only transfer important information. Overall, these results paint a promising picture: pruning teachers improves distillation if the proper distillation technique is used.

KD-distilled Pubmed models outperform the teacher baseline. Another interesting phenomena happens for the Pubmed dataset: almost all distilled models perform better than the *teacher* baseline; this may be due to the fact that pruned teachers for the Pubmed dataset are significantly better than their unpruned counterpart, as shown in Figure 1.

The LSP loss does not improve performance. The performance drop for models distilled with the LSP loss may be explained by the fact that the LSP loss tries to make the latent features in the student network match the latent features of the teacher; however, a heavily pruned teacher will have most of these latent features missing - this may lead to a student trying to "prune itself".



(a) Distillation was performed with the logit-based KD loss.

(b) Distillation was performed with the LSP loss.

Figure 2: Test accuracies of distilled models w.r.t. to the sparsity of their teacher. For each distilled model I report its test accuracy mean and std. The student baseline is a model trained without distillation. The teacher baseline is the same as in Figure 1.

5 Conclusion

This project investigated the viability of using pruning to improve knowledge distillation in GAT networks. The results indicate that we can build "student-friendly" teachers for transductive node

Table 4: Performance of students trained with pruned teachers for different sparsity levels. The arrows indicate the model's improvement to the model trained with an unpruned teacher. The KD and LSP columns indicate the loss function the teacher was distilled with.

Dataset	Teacher sparsity (%)		Teacher accuracy (%)	Student accuracy (%)		
	Graph	Model		KD	LSP	Baseline
Cora	0	0	81.0	$78.87_{\pm 0.896}$	$76.70_{\pm 1.445}$	$76.50_{\pm 0.572}$
	30.17	79.13	81.6	$79.97_{\pm 0.170}(\uparrow)$	$76.80_{\pm 1.158}(\uparrow)$	
	33.66	83.22	78.9	$79.27_{\pm 0.125}(\uparrow)$	$76.33_{\pm 0.984}(\downarrow)$	
	62.26	98.56	80.5	$79.37_{\pm 0.776}(\uparrow)$	$76.20_{\pm 0.898}(\downarrow)$	
Citeseer	0	0	70.9	$66.80_{\pm 0.327}$	$67.07_{\pm 0.826}$	$65.67_{\pm 1.558}$
	30.17	79.13	70.6	$69.07_{\pm 1.078}(\uparrow)$	$67.37_{\pm 0.519}(\uparrow)$	
	33.66	83.22	70.6	$67.37_{\pm 2.243}(\uparrow)$	$66.77_{\pm 0.858}(\downarrow)$	
	62.26	98.56	70.4	$68.13_{\pm 0.974}(\uparrow)$	$67.60_{\pm 1.098}(\uparrow)$	
Pubmed	0	0	76.2	$76.93_{\pm 0.309}$	$74.60_{\pm 0.748}$	$75.07_{\pm 0.660}$
	30.17	79.13	78.7	$77.00_{\pm 0.245}(\uparrow)$	$74.93_{\pm 0.834}(\uparrow)$	
	33.66	83.22	78.6	$77.00_{\pm 0.082}(\uparrow)$	$74.37_{\pm 0.419}(\downarrow)$	
	62.26	98.56	80.6	$76.37_{\pm 0.478}(\downarrow)$	$74.57_{\pm 0.450}(\downarrow)$	

classification tasks by performing iterative pruning of the teacher GAT model and its associated graph. The experiments show that such teachers are effective even when they have high sparsity levels: at 98.55% model sparsity and 64.15% graph sparsity, we can still see some improvements over training with unpruned teachers. This is particularly important for transductive node classification tasks on huge graphs, since this sparsity may reduce the amount of memory necessary when training on such datasets. However, I discuss some of the main limitations of this project below.

Limited scope. This project has a limited scope along three dimensions; firstly, the experiments are run on a single type of GNN network. While GAT networks are widely used in the GNN community, further experiments should be run on other popular networks, such as GCNs [Kipf and Welling, 2016b] and GINs [Xu et al., 2018]. Secondly, I investigated the performance of "student-friendly" teachers only on the task of transductive node classification. Similar work should again check the hypothesis of the pruned teacher on other GNN tasks, such as link prediction and graph classification. More importantly, an investigation should focus on the best pruning strategy for *inductive* tasks, where datasets contain more than one graph, some of which are never seen during training. Lastly, due to limited computing resources, this project has used three popular citation networks as datasets; however, their small size may make them non-representative of the performance of the **prune-then-distill** pipeline for large graphs. I hope, however, that this project may act as a proof-of-concept for future development and deployment of efficient GNNs.

Graph distillation and pruning. Additionally, graph pruning and distillation are open areas of research in the GNN community, as discussed in Sections 2.2 and 2.3. Some pruning and distillation strategies work better for some tasks than others; as of today, there is no clear-cut winner when it comes to optimising the size and performance of GNNs. I have left out many other possible distillation techniques, such as G-CRD [Joshi et al., 2022] and FitNet [Romero et al., 2014]. While my results indicate that LSP distillation does not improve the performance of student models, other techniques might outperform classic logit-based KD.

Unstructured pruning. This project investigates only the impact of unstructured pruning over the distillation pipeline. This type of pruning, while reducing model size, makes models hard to optimise using accelerated hardware. While there are some libraries that optimise sparse operations, the full impact of unstructured pruning is hard to be taken advantage of by modern hardware. Another line of future work could focus on re-running my proposed pipeline while performing structured pruning.

References

Tianlong Chen, Yongduo Sui, Xuxi Chen, Aston Zhang, and Zhangyang Wang. A unified lottery ticket hypothesis for graph neural networks, 2021. URL https://arxiv.org/abs/2102.06790.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. 2018. doi: 10.48550/ARXIV.1803.03635. URL https://arxiv.org/abs/1803.03635.

C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. Citeseer: An automatic citation indexing system. In *Proceedings of the Third ACM Conference on Digital Libraries*, DL '98, page 89–98, New York, NY, USA, 1998. Association for Computing Machinery. ISBN 0897919653. doi: 10.1145/276675.276685. URL https://doi.org/10.1145/276675.276685.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

Babak Hassibi and David Stork. Second order derivatives for network pruning: Optimal brain surgeon. *Advances in neural information processing systems*, 5, 1992.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. URL https://arxiv.org/abs/1503.02531.

Chaitanya K Joshi, Fayao Liu, Xu Xun, Jie Lin, and Chuan Sheng Foo. On representation knowledge distillation for graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. URL https://arxiv.org/abs/1412.6980.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2016a. URL https://arxiv.org/abs/1609.02907.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016b.

Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016.

Chuang Liu, Xueqi Ma, Yibing Zhan, Liang Ding, Dapeng Tao, Bo Du, Wenbin Hu, and Danilo Mandic. Comprehensive graph gradual pruning for sparse training in graph neural networks, 2022. URL https://arxiv.org/abs/2207.08629.

Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5191–5198, 2020.

Dae Young Park, Moon-Hyun Cha, Daesin Kim, Bohyung Han, et al. Learning student-friendly teacher networks for knowledge distillation. *Advances in Neural Information Processing Systems*, 34:13292–13303, 2021.

Jinhyuk Park and Albert No. Prune your model before distill it. In *European Conference on Computer Vision*, pages 120–136. Springer, 2022.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets, 2014. URL https://arxiv.org/abs/1412.6550.

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.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation, 2019. URL https://arxiv.org/abs/1910.10699.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Yulong Wang, Xiaolu Zhang, Lingxi Xie, Jun Zhou, Hang Su, Bo Zhang, and Xiaolin Hu. Pruning from scratch. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12273–12280, 2020.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks?, 2018. URL https://arxiv.org/abs/1810.00826.

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.

Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.

Marinka Zitnik and Jure Leskovec. Predicting multicellular function through multi-layer tissue networks. *Bioinformatics*, 33(14):i190–i198, 2017.