MODEL-TO-DATA DISTILLATION FOR GRAPH NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph neural networks (GNNs) increasingly rely on sophisticated architectures and training procedures to achieve desirable properties such as high predictive performance, fairness, and robustness. However, these properties typically remain tied to the models that learn them, limiting their transferability to simpler models and downstream settings. We introduce *model-to-data (M2D) distillation*, a new distillation paradigm that transfers properties learned by a complex GNN teacher into graph data, enabling simpler models to recover them through standard training. M2D distillation explicitly trades model complexity for data complexity by jointly learning augmented node features and graph structure that encode the teacher's behavior. The resulting graph serves as a persistent medium for knowledge transfer and can be used with different downstream models. We show that M2D distillation enables simple GNNs to approximate the behavior of substantially more sophisticated teachers, including fairness-aware GNNs, Graph Attention Networks, and Graph Transformers, while maintaining comparable predictive performance and transferring desirable properties of the teacher.

1 INTRODUCTION

Graph Neural Networks (GNNs) have achieved state-of-the-art performance across a wide range of graph learning tasks, including node classification, link prediction, and graph classification (Gilmer et al., 2017; Kipf & Welling, 2017; Veličković et al., 2018; Hamilton et al., 2017). Alongside these advances, increasingly sophisticated models and training procedures have been developed to endow GNNs with desirable properties such as greater expressive power (Xu et al., 2019), fairness (Dai & Wang, 2021), robustness (Zügner et al., 2018), calibration (Wang et al., 2021), privacy (Sajadmanesh et al., 2023), etc. Yet these properties typically remain tied to the specialized models and procedures used to obtain them. Such models may be inaccessible to researchers and developers due to intellectual property restrictions or impractical to deploy because of their computational resource requirements. This contrasts sharply with training data, which are persistent and model-agnostic: the same dataset can be reused to train generations of models with different architectures, objectives, and computational budgets. As a result, while data are often accessible and readily reusable, the desirable properties (e.g., expressive power, fairness) learned from them are not.

In this work, we ask whether properties learned by a GNN can themselves be transferred through data. Consider, for example, a Graph Convolutional Network (GCN) trained using a specialized fairness-aware procedure (Dai & Wang, 2021). Rather than distilling this model directly into a particular student, can we use the teacher to modify the training data such that a vanilla GCN (Kipf & Welling, 2017) trained on the same data is also fair? Our example is illustrated in Figure 1. More generally, given a teacher GNN model exhibiting a desired property, can we transform its training data so that new models inherit this property without requiring access to the teacher, its original training procedure, or an explicit property-specific objective?

We formulate this problem as *model-to-data (M2D) distillation*. Unlike conventional knowledge distillation, which transfers knowledge from a teacher model directly to a particular student, M2D uses data as the medium through which learned properties are transferred. We position M2D within a unified model–data complexity space, illustrated in Figure 2. Knowledge (K) distillation (Gou et al., 2021) reduces model complexity while approximately preserving learned behavior, corresponding to a vertical shift in this space, whereas Data (D) distillation (Lei & Tao, 2023) reduces data complexity

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

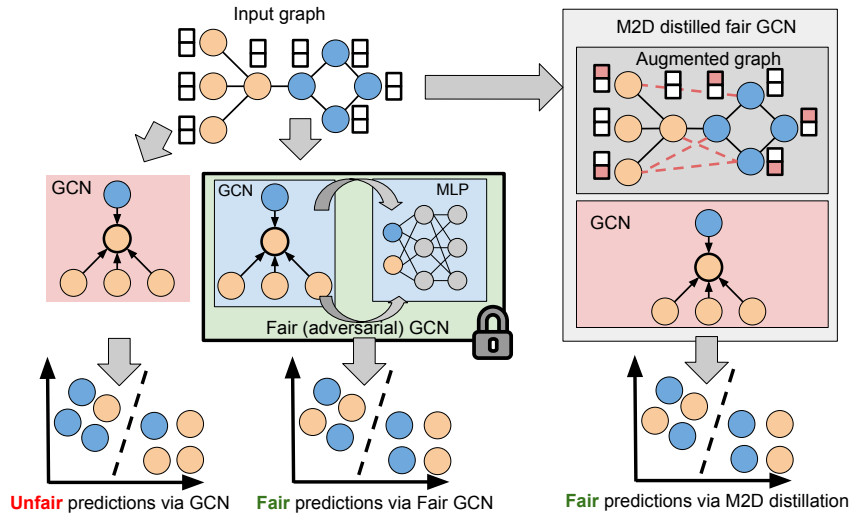

Figure 1: Model-to-data distillation (M2D) from a fair Graph Convolution Network (GCN) to a vanilla GCN. M2D distills the fair GCN into an augmented graph where features and structure are modified. Intuitively, the augmentation should remove the bias from the graph.

while preserving the ability to train effective models, corresponding to a horizontal shift. In contrast, M2D moves in the opposite direction: it trades model complexity for data complexity by transferring properties encoded in a sophisticated teacher into the training data, enabling simpler models to recover those properties through standard training.

M2D is particularly useful in resource-constrained settings, where deploying the original teacher may be infeasible but training or deploying lightweight models is practical. Moreover, because the distilled property is mediated by the data, it can serve as a persistent form of behavioral knowledge that is reused across architectures and future generations of models. M2D can also provide a new mechanism for model interpretation and data auditing. By examining how a teacher modifies the original dataset to make a desired property emerge under standard training, we can identify features or structural patterns responsible for undesirable model behavior.

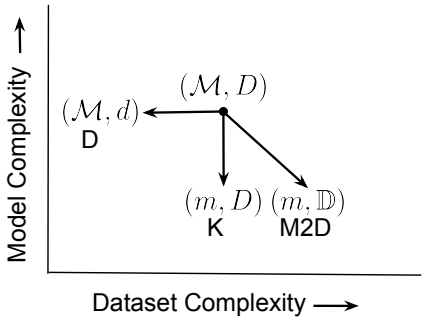

Figure 2: Distillation design space with data (x-axis) and model (y-axis) complexity. Knowledge (K) distillation reduces model complexity (vertical), and data (D) distillation reduces data complexity (horizontal). Model-to-Data (M2D) distillation transfers capacity from the model to data (diagonal).

We summarize our contributions as follows: (1) we introduce *model-to-data (M2D) distillation*, a general framework for trading model complexity for data complexity by transferring desirable properties from sophisticated GNNs into graph data; (2) we instantiate M2D through an iterative framework that combines feature and structure learning with multi-objective optimization to encode teacher properties into the input graph; and (3) we empirically demonstrate that M2D enables simpler GNNs trained with standard procedures to approximate properties of more sophisticated teachers, including fairness-aware GNNs, Graph Attention Networks, and Graph Transformers.

1.1 Related Work

Graph augmentation. Graph augmentation modifies graph structure or node features to improve GNN performance (Zhou et al., 2023; Li et al., 2023; Zhu et al., 2021; Attali et al., 2024). While many approaches perturb graph topology by adding or removing nodes and edges (Franceschi et al., 2019; Chen et al., 2020; Fatemi et al., 2021; Zhao et al., 2022; Rong et al., 2020; Gasteiger et al., 2019; Dong & Kluger, 2023), some methods operate in feature space (You et al., 2020; Jeong et al., 2024; Guo & Mao, 2023). For instance, FLAG learns adversarial perturbations of node features to

improve robustness and generalization (Kong et al., 2022), while graph contrastive learning methods commonly construct augmented views through feature masking or corruption (You et al., 2020).

Data distillation on graphs. Data distillation on graphs is commonly referred to as graph condensation (GC), which seeks to construct compact graphs that preserve the utility of the original data (Hashemi et al., 2024). GC methods can be broadly categorized according to their primary objectives. Most methods focus on performance preservation, learning condensed graphs such that GNNs trained on them achieve performance comparable to models trained on the original graph (e.g., GCond, SFGC, CTRL, OpenGC, GCSR, and KiDD) (Zhang et al., 2024; Gao et al., 2024b; Zheng et al., 2023; Liu et al., 2024b; Jin et al., 2022b; Xu et al., 2023). Other methods emphasize generalization, constructing condensed graphs that transfer across architectures or tasks by preserving salient structural and feature information (e.g., GDEM and OpenGC) (Liu et al., 2024a; Gao et al., 2024b). A complementary line of work focuses on *efficiency*, accelerating the condensation process through more efficient encoding, optimization, or synthetic-graph generation (e.g., DosCond, CaT, and EXGC) (Jin et al., 2022a; Gao et al., 2024a; Fang et al., 2024). Recent approaches have extended GC beyond predictive performance to properties such as fairness, with FGD and GCARe incorporating fairness constraints or regularization into the condensation process (Feng et al., 2023; Mao et al., 2023), and robustness, with RobGC filtering noisy information while preserving core causal information for reliable downstream prediction (Gao et al., 2025).

Knowledge distillation on graphs. Existing methods distill (1) output knowledge by aligning teacher and student predictions (e.g., TinyGNN, GFKD, DFAD-GNN, KDGA, CPF, and GLNN) (Yan et al., 2020; Deng & Zhang, 2021; Zhuang et al., 2022; Wu et al., 2022; Yang et al., 2021; Zhang et al., 2022b); (2) structural knowledge by preserving local or global relationships encoded by the teacher (e.g., LSP, FreeKD, AlignAhead, GNN-SD, CKD, G-CRD, ROD, MSKD, Cold Brew, GKD, and NOSMOG) (Yang et al., 2020; Feng et al., 2022; Guo et al., 2022; Chen et al., 2021; Wang et al., 2022a; Joshi et al., 2022; Zhang et al., 2021; 2022a; Zheng et al., 2022; Yang et al., 2022; Tian et al., 2023); or (3) representation knowledge by aligning intermediate embeddings across models or layers (e.g., T2-GNN, and GraphAKD) (Huo et al., 2023; He et al., 2022). Some approaches combine distillation with augmentation by modifying structure, features, or graph views before or during knowledge transfer (e.g., KDGA, T2-GNN, NOSMOG, GAFD, and GraphGR) (Wang et al., 2025; Lee et al., 2025; Wu et al., 2022; Huo et al., 2023; Tian et al., 2023).

M2D differs from conventional graph augmentation by distilling teacher behavior, rather than optimizing a predefined downstream objective. Moreover, our approach differs from knowledge and data distillation by transferring the teacher's behavior into an augmented graph rather than into a student model or compressed dataset. We note that M2D does not apply graph augmentation to simply facilitate teacher-to-student knowledge transfer, while the distilled knowledge ultimately resides in the student model. Instead, M2D reverses this role and treats the augmented graph itself as the target of distillation. Teacher behavior is encoded into its transformed features and structure, allowing the resulting graph to serve as a persistent, model-agnostic medium from which independently trained downstream models can recover the teacher's behavior without further access to the teacher.

2 Model to Data (M2D) Distillation

2.1 Problem Definition

We propose Model-to-Data Distillation (M2D), a new distillation paradigm that transfers complexity from a Graph Neural Network (GNN) model to the data. M2D enriches the data so that a lightweight GNN can achieve performance comparable to a more complex one (see Figure 2).

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ be an undirected graph where $\mathcal{V}$ is the set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, and $\mathbf{X} \in \mathbb{R}^{n \times d}$ are the node attributes. The matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is the adjacency matrix of $\mathcal{G}$ where $\mathbf{A}_{uv} = 1$ if there is an edge between u and v and $\mathbf{A}_{uv} = 0$, otherwise. Let f_T be the teacher model with parameter θ_T and f_s be a student model with parameter θ_s where $|\theta_T| > |\theta_s|$. Moreover, let the prediction of the teacher model and the student model be $\hat{\mathbf{y}}_T$ and $\hat{\mathbf{y}}_s$, respectively.

M2D learns a transformation f_g with parameters θ_g that enriches the original graph data $\mathcal{G}$ by modifying the graph structure or the node features, yielding $\tilde{\mathcal{G}} = (\mathcal{V}, \tilde{\mathcal{E}}, \tilde{\mathbf{X}})$. We learn the graph transformation f_g and the student model f_s jointly by optimizing the following bilevel objective:

$$\min_{\theta_g} \quad \mathcal{L}_{\text{dis}}\Big(f_T(\mathcal{G}), f_s(\tilde{\mathcal{G}}; \theta_s^*)\Big) - \mathcal{S}\Big(\mathcal{G}, \tilde{\mathcal{G}}\Big) \quad \text{s.t.} \quad \theta_s^* = \arg\min_{\theta_s} \mathcal{L}_{\text{cls}}\Big(f_s(\tilde{\mathcal{G}}), \mathbf{y}\Big). \tag{1}$$

where $\tilde{\mathcal{G}} = f_g(\mathcal{G}; \theta_g)$ is the augmented graph, $\mathcal{L}_{\text{dis}}$ measures the difference between the outputs of the teacher f_T and student f_s, $\mathcal{L}_{\text{cls}}$ is the classification loss between true labels $\mathbf{y}$ and student output, and $\mathcal{S}$ is a function that computes the similarity between $\mathcal{G}$ and $\tilde{\mathcal{G}}$. The term $\mathcal{S}(\mathcal{G}, \tilde{\mathcal{G}})$ enforces that the model knowledge is distilled into the data while introducing only minimal modifications to the original graph. In general, both the teacher model f_T and the student model f_s can be arbitrary architectures. The framework can also be generalized for a family of student models. In this work, however, we restrict our focus to GNNs and Graph Transformers (GTs) as teacher models and GNNs and MLPs as student models. Our framework assumes only partial access to teacher's logits. Let $\mathcal{V}_{\text{tr}} \in \mathcal{V}$ denote the set of nodes for which both teacher logits and ground-truth labels are available.

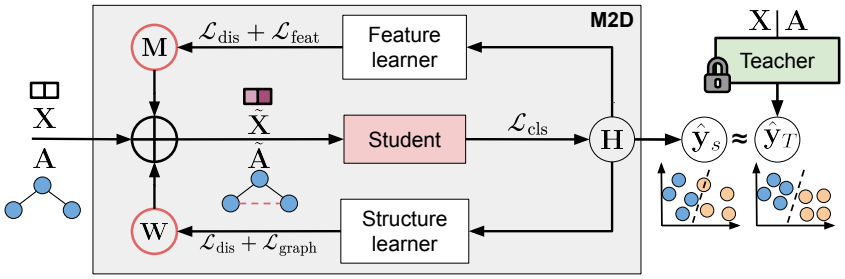

Figure 3: The feature and structure learners iteratively refine features and structure by optimizing $\mathcal{L}_{\text{dis}}$, and $\mathcal{L}_{\text{dis}}, \mathcal{L}_{\text{graph}}$, respectively. The student model is then trained on the learned graph, optimizing $\mathcal{L}_{\text{cls}}$ to align with the teacher while improving predictive performance.

2.2 Proposed Model

We propose a solution to the M2D optimization problem of Eq 1. The learning process alternates between (i) updating the graph topology and node features using current embeddings and (ii) updating embeddings using the augmented graph. Let the student GNN f_s generate node embedding $\mathbf{H}_s \in \mathbb{R}^{n \times d_h}$. We decompose the transformation f_g into two components: a feature Learner f_ϕ that learns new node features and a structure learner f_a that learns new adjacency weights. Figure 3 shows the flow diagram of the proposed model.

2.3 Learning node features and graph structure

Feature Learner: Let $f_\phi : \mathbb{R}^{d_h} \to \mathbb{R}^d$ be a learnable feature masking function that generates a node-specific continuous feature mask from the student node embeddings. At iteration t, the mask is computed from node embeddings $\mathbf{H}_s$ at time $t-1$ as

$$\mathbf{M}^t = \sigma\big(f_\phi(\mathbf{H}_s^{t-1})\big) \in [0,1]^{n \times d},$$

where $\sigma(\cdot)$ denotes the sigmoid function. The mask assigns an importance weight to each feature of every node. The augmented feature matrix is then obtained through element-wise multiplication,

$$\tilde{\mathbf{X}}^t = \mathbf{X} \odot \mathbf{M}^t \in \mathbb{R}^{n \times d},$$

where $\odot$ is the Hadamard product. Initial student node embeddings are obtained as $\mathbf{H}_s^0 = f_s(\mathbf{X}, \mathbf{A})$.

Structure Learner: We learn a new graph structure by predicting pairwise node relationships from their embeddings. Given node representations $\mathbf{H}_s^{t-1}$, let $f_a : \mathbb{R}^{2d_h} \to [0,1]$ be a function that learns a new weighted adjacency matrix. The learned matrix is defined element-wise as:

$$\mathbf{W}_{uv}^t = \sigma\big(f_a(\mathbf{h}_{s,u}^{t-1}, \mathbf{h}_{s,v}^{t-1})\big), \quad \mathbf{W}^t \in [0,1]^{n \times n},$$

where $\mathbf{h}_{s,u}^{t-1}$ and $\mathbf{h}_{s,v}^{t-1}$ are embeddings of node u and v at time $t-1$. For directed graphs, f_a operates on the concatenation of node embeddings:

$$\mathbf{W}_{uv}^t = \sigma\big(f_a([\mathbf{h}_{s,u}^{t-1} \,\|\, \mathbf{h}_{s,v}^{t-1}])\big).$$

For undirected graphs, we enforce permutation invariance by symmetrizing the input:

$$\mathbf{W}_{uv}^t = \sigma\big(f_a(\mathbf{h}_{s,u}^{t-1} + \mathbf{h}_{s,v}^{t-1} \parallel |\mathbf{h}_{s,u}^{t-1} - \mathbf{h}_{s,v}^{t-1}|)\big),$$

which ensures $\mathbf{W}_{uv}^t = \mathbf{W}_{vu}^t$. The final weighted adjacency matrix at iteration t is a combination of the original graph with trained and untrained weights:

$$\tilde{\mathbf{A}}_{uv}^t = (1-\gamma)\mathbf{A}_{uv} + \gamma((1-\beta)\mathbf{W}_{uv}^t + \beta s(\mathbf{x}_u, \mathbf{x}_v)),$$

where γ and β are hyperparameters, and $s(\mathbf{x}_u, \mathbf{x}_v)$ is the cosine similarity between u and v based on original features. We then use $\tilde{\mathbf{X}}^t$ and $\tilde{\mathbf{A}}^t$ to generate representations $\mathbf{H}_s^t = f_s(\tilde{\mathbf{X}}^t, \tilde{\mathbf{A}}^t)$ at iteration t. The final graph is generated either when $\mathcal{L}_{\text{cls}}$ converges or after a fixed number of iterations.

2.4 Training Objective

Let $\tilde{\mathcal{G}}^t = (\mathcal{V}, \tilde{\mathbf{A}}^t, \tilde{\mathbf{X}}^t)$ be the augmented graph at time t. The student encoder f_s produces node representations $\mathbf{H}_s^t = f_s(\tilde{\mathbf{A}}^t, \tilde{\mathbf{X}}^t)$, which are mapped to class logits $\hat{\mathbf{y}}_s^t = f_c(\mathbf{H}_s^t)$ using a classifier f_c. We partition the model parameters into groups $\Theta_s = \{\theta_s, \theta_c\}$ and $\Theta_g = \{\theta_\phi, \theta_a\}$, where Θ_s denotes the student encoder and classifier parameters and Θ_g denotes the graph augmentation parameters.

Classification loss $\mathcal{L}_{\text{cls}}$: We use the standard supervised cross-entropy loss.

$$\mathcal{L}_{\text{cls}} = -\frac{1}{|\mathcal{V}_{\text{tr}}|} \sum_{u \in \mathcal{V}_{\text{tr}}} \mathbf{y}_u^\top \log \hat{\mathbf{y}}_{s,u}^t. \tag{2}$$

Distillation loss $\mathcal{L}_{\text{dis}}$: We align student and teacher predictive distributions using temperature-scaled soft targets via the Kullback–Leibler (KL) Divergence. Let the teacher's logits be $\hat{\mathbf{y}}_T$, then:

$$\mathcal{L}_{\text{dis}} = \tau^2 \frac{1}{|\mathcal{V}_{\text{tr}}|} \sum_{u \in \mathcal{V}_{\text{tr}}} D_{\text{KL}} \left(\text{softmax}\left(\frac{\hat{\mathbf{y}}_{T,u}}{\tau}\right), \text{softmax}\left(\frac{\hat{\mathbf{y}}_{s,u}^t}{\tau}\right)\right). \tag{3}$$

where τ smooths the distributions, exposing inter-class similarity and teacher uncertainty not captured by teacher logits (Hinton et al., 2015). The τ^2 term preserves gradient magnitudes.

Feature reconstruction loss $\mathcal{L}_{\text{feat}}$: To ensure the learned mask does not drastically change the original features, we calculate the feature reconstruction loss $\mathcal{L}_{\text{feat}}$ as follows:

$$\mathcal{L}_{\text{feat}} = ||\mathbf{X} - \tilde{\mathbf{X}}^t||_F^2 \tag{4}$$

where $||.||_F$ is the Frobenius norm.

Graph regularization loss $\mathcal{L}_{\text{graph}}$: Following Kalofolias (2016), we impose additional constraints on $\mathbf{W}^t$ through our graph structure loss $\mathcal{L}_{\text{graph}}$:

$$\mathcal{L}_{\text{graph}} = -\frac{\lambda_{\text{deg}}}{n} \mathbf{1}^T \log(\mathbf{W}^t \mathbf{1}) + \frac{\lambda_{\text{sparse}}}{n^2} ||\mathbf{W}^t||_F^2. \tag{5}$$

where the first term penalizes zero-degree nodes and the second term enforces sparsity on the learned adjacency, encouraging the augmented graph to remain close to the original topology.

The feature reconstruction loss and structural regularization are optimized only with respect to the graph augmentation parameters Θ_g. Therefore, their gradients with respect to the parameters Θ_s are constrained to vanish: $\nabla_{\Theta_s} \mathcal{L}_{\text{feat}} = 0$ and $\nabla_{\Theta_s} \mathcal{L}_{\text{graph}} = 0$. At iteration t, the student parameters Θ_s are optimized using the classification loss, whereas the graph modifier parameters Θ_g are optimized using the distillation loss and the structural regularization term:

$$\Theta_s^* = \arg\min_{\Theta_s} \mathcal{L}_{\text{cls}}, \quad \Theta_g^* = \arg\min_{\Theta_g} \mathcal{L}_{\text{dis}} + \alpha \mathcal{L}_{\text{feat}} + \mathcal{L}_{\text{graph}}.$$

where α is a hyperparameter that regulates the impact of the feature reconstruction loss. Importantly, the graph modifier is not optimized using the classification loss. Its learning signal comes exclusively from the teacher–student distillation objective, ensuring that the augmented graph is explicitly guided by the teacher's behavior.

The M2D training algorithm and computational complexity are detailed in the Appendix.

3 THEORETICAL ANALYSIS

Theorem 1 shows that the fairness gap between a downstream model and the teacher is bounded by the distillation error and by how closely the original student and downstream model, when trained with cross-entropy, approximate the optimal predictor on the transformed graph. Theorem 2 shows that M2D transfers the teacher's attention-weighted neighborhood information into both learned feature masks and edge weights, up to residuals from local linearization and imperfect distillation.

Theorem 1. *Let $\tilde{\mathcal{G}}$ be the graph learned by M2D, and let $\hat{Y}_T$ and $\hat{Y}_s$ denote the teacher and M2D student predictions. For binary sensitive attribute $a \in \{0,1\}$, define $\mathrm{DP}(f) = \left|\mathbb{E}[\hat{Y}_f \mid a=0] - \mathbb{E}[\hat{Y}_f \mid a=1]\right|$. Let κ_g denote the teacher–student KL divergence and $\xi_g(f)$ the excess log-loss of model f, both conditioned on $a=g$. Then any independently trained downstream student $f_{s'}$ on $\tilde{\mathcal{G}}$ satisfies $|\mathrm{DP}(f_{s'}) - \mathrm{DP}(f_T)| \le \sum_{g\in\{0,1\}} \left(\sqrt{\frac{\kappa_g}{2}} + \sqrt{\frac{\xi_g(f_s)}{2}} + \sqrt{\frac{\xi_g(f_{s'})}{2}}\right)$.*

Theorem 2. *Let the teacher be a GAT or GT with attention coefficients α_{ij} and let the teacher and student have linear prediction heads. Assume a local first-order approximation of the teacher representation, finite teacher logits, and a minimum-ℓ_2-norm student solution. Then there exists a linear transformation Q and residual $\mathbf{r}_i$ such that $\mathbf{m}_i = \sigma(f_\phi(P_i))$, and $w_{ij} = \sigma\left(f_a(P_i + P_j, |P_i - P_j|)\right)$, where $P_i = \sum_{k\in\mathcal{N}(i)} \alpha_{ik} Q\mathbf{x}_k + \mathbf{r}_i$.*

4 EXPERIMENTAL EVALUATION

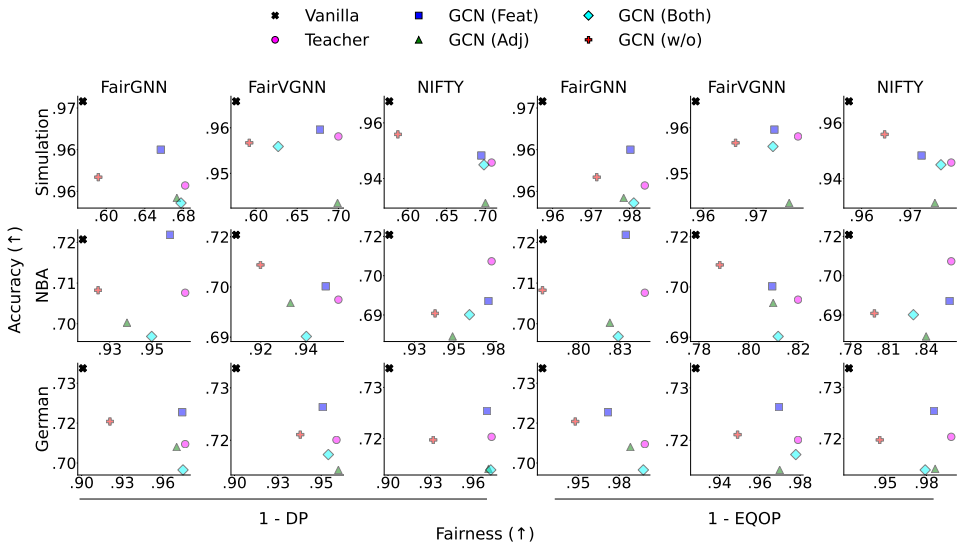

Figure 4: Accuracy–fairness trade-offs for FairGNN, FairVGNN, and NIFTY and their distilled variants. The y-axis shows accuracy, while the x-axis reports 1 - DP and 1 - EQOP. Points closer to the top-right indicate a better trade-off. Results show that bias in NBA is primarily feature-driven, whereas SIMULATION and GERMAN datasets exhibit bias in both features and topology.

We evaluate how M2D transfers teacher knowledge from model to data across fair GNNs, Graph Attention Networks (GATs) (Veličković et al., 2018), and Graph Transformers (GTs). Our analysis focuses on how model complexity and fairness properties are distilled into the data. We consider three variants of M2D: feature-only (Feat.), adjacency-only (Adj), and joint feature–adjacency (Both). We additionally evaluate a baseline where the teacher is distilled into the student without M2D (w/o) and a vanilla GCN with no distillation (Van). Both the feature learner and the adjacency learner are implemented as MLPs. Additional details about the experimental setup are in the Appendix.

4.1 FAIR GNN ANALYSIS

We apply M2D to three fair GNNs (FairGNN, FairVGNN, and NIFTY) (Dai & Wang, 2021; Wang et al., 2022b; Agarwal et al., 2021) as teacher models, with a GCN as the student. We evaluate utility

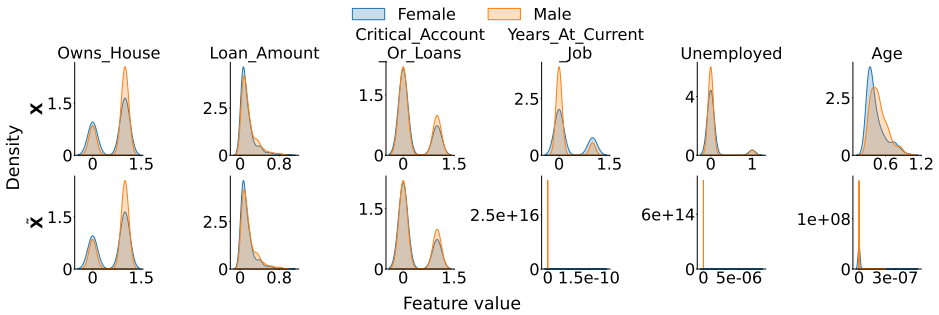

Figure 5: KDE plots for six original ($\mathbf{X}$) and learned ($\tilde{\mathbf{X}}$) features across sensitive groups on the GERMAN dataset with GCN (Feat) as the student and FairGNN as the teacher. Features in $\tilde{\mathbf{X}}$ concentrated around zero correspond to those whose contributions are suppressed by the learned masks.

via accuracy and fairness using Demographic Parity (DP) and Equal Opportunity (EQOP). Details on datasets, baselines, hyperparameters, and evaluation are provided in the Appendix.

Figure 4 shows the accuracy–fairness trade-off for each model and its distilled variants. The y-axis reports accuracy, while the x-axis shows $1-\text{DP}$ and $1-\text{EQOP}$. Points closer to the top-right indicate a better trade-off. In the NBA dataset, bias is mainly feature-driven, as fairness improves with feature-only distillation. In contrast, the SIMULATION and the GERMAN datasets exhibit bias in both features and structure, requiring joint feature–adjacency distillation.

Figure 5 shows Kernel Density Estimation (KDE) plots for six features from the GERMAN dataset, with FairGNN as the teacher and GCN (Feat.) as the student. The first row shows the original features $\mathbf{X}$ and the second row shows the corresponding learned features $\tilde{\mathbf{X}}$. We select three features that exhibit substantial changes after masking and three that remain unchanged. For the modified features, values concentrated around zero indicate that their contributions are suppressed by the learned masks, illustrating how M2D selectively modifies feature contributions to transfer the teacher's fairness behavior. Analysis of the remaining features and the modified adjacency for the GCN (Adj) student is provided in the Appendix.

4.2 GAT AND GT ANALYSIS

More complex architectures, such as GAT and GTs, are often preferred over simpler models like GCN due to their superior performance. We analyze the advantage of GAT and Graphormer (Ying et al., 2021) over GCN and MLP (see Appendix) using our approach. Table 1 reports node classification accuracy across multiple datasets, comparing teacher models (GAT and GT) against the GCN student model under different distillation variants.

		Teacher	GCN (Van.)	GCN (w/o)	GCN (Feat.)	GCN (Adj)	GCN (Both)
CORA	GAT	83.24 ± 0.28	79.23 ± 0.26	79.27 ± 0.13	$\mathbf{82.52 \pm 0.44}$	81.10 ± 0.40	82.05 ± 0.41
	GT	77.42 ± 0.12		76.19 ± 0.14	77.13 ± 0.21	76.14 ± 0.12	76.01 ± 0.12
CITESEER	GAT	73.21 ± 0.21	71.64 ± 0.16	71.62 ± 0.27	$\mathbf{72.04 \pm 0.13}$	71.98 ± 0.13	72.02 ± 0.12
	GT	65.28 ± 0.11		64.93 ± 0.21	66.20 ± 0.21	65.11 ± 0.19	65.02 ± 0.11
PHOTO	GAT	91.35 ± 0.32	90.23 ± 0.23	90.01 ± 0.21	$\mathbf{90.91 \pm 0.21}$	90.42 ± 0.20	91.27 ± 0.19
	GT	94.20 ± 0.16		90.08 ± 0.14	92.88 ± 0.14	92.17 ± 0.18	$\mathbf{92.89 \pm 0.11}$
CORNELL	GAT	53.67 ± 0.41	53.61 ± 0.41	53.15 ± 0.19	54.10 ± 0.21	53.64 ± 0.20	$\mathbf{54.13 \pm 0.11}$
	GT	70.59 ± 0.12		52.98 ± 0.21	58.11 ± 0.18	56.21 ± 0.11	$\mathbf{58.79 \pm 0.11}$
TEXAS	GAT	64.31 ± 0.24	62.36 ± 0.20	62.71 ± 0.13	62.69 ± 0.12	64.32 ± 0.14	$\mathbf{64.49 \pm 0.15}$
	GT	73.89 ± 0.14		62.39 ± 0.31	65.16 ± 0.11	65.59 ± 0.11	$\mathbf{65.56 \pm 0.11}$

Table 1: Node classification accuracy comparing teacher models (GAT, GT) against a student GCN under different settings. 'Van' is the student without any distillation, 'w/o' is the standard knowledge distillation (without M2D), and 'Feat.', 'Adj' and 'Both' are M2D applied to features, adjacency, and both, respectively. Results show that M2D, particularly Feat. and Both consistently improve student performance and can match teacher accuracy across datasets.

M2D generally improves transfer when the teacher provides useful knowledge, while negative transfer can occur when the teacher is weaker than the student. Feature-based distillation provides the

most reliable gains, while adjacency-only distillation is less effective. Joint distillation (both) typically achieves the second-best results, in some cases matching the teacher. These findings demonstrate that M2D can to effectively transfer knowledge from more complex architectures to the data.

Motivated by Theorem 2, we empirically investigate whether the teacher's attention patterns are reflected in the similarity structure of the learned feature masks. Figure 6 analyzes how the learned feature mask aligns with the attention mechanism of a teacher GAT model. We first compute pairwise similarity between node representations using Gaussian similarity based on their Euclidean distance, denoted by, $\mathrm{sim}(\mathbf{m}_i, \mathbf{m}_j)$ and group node pairs into bins based on their similarity. For each bin, we measure the average attention weight of the teacher's existing edges. For homophilic datasets (CORA, CITESEER, PHOTO), we observe a clear increasing trend: higher similarity corresponds to larger attention. In heterophilic datasets (CORNELL, TEXAS), a weaker but still positive relationship is observed. This difference highlights that the alignment between feature-mask similarity and attention depends on the underlying graph. The analysis of the learned feature mask with respect to the GT attention is in the Appendix.

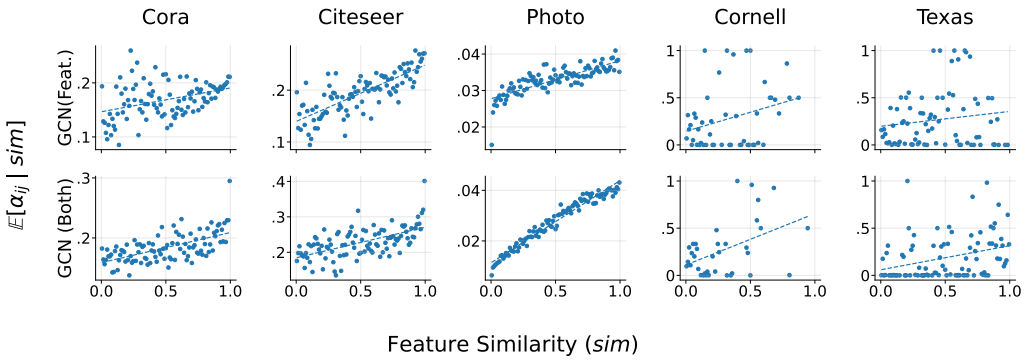

Figure 6: Analysis of the learned feature mask $\mathbf{M}$ with respect to the attention of the teacher GAT. For each dataset, we partition the range of mask similarity $\mathrm{sim}(\mathbf{m}_i, \mathbf{m}_j)$ for existing edges into bins and compute the average attention within each bin. The increasing trend in homophilic datasets (CORA, CITESEER, PHOTO) indicates that higher attention weights correspond to higher mask similarity. We can also see the trend in heterophilic datasets (CORNELL, TEXAS).

Figure 7 visualizes edge importance on CORA (homophilic) using row-normalized M2D edge weights and GAT attention coefficients (Layer 1). Edge color and line width reflect magnitude (darker, thicker edges indicating higher values). The M2D weights are normalized per node (rows sum to one), enabling a comparison with attention coefficients. Using an identical layout, both visualizations exhibit highly similar patterns, indicating that the learned edge weights closely match the GAT attention mechanism. Corresponding results on the heterophilic CORNELL dataset, together with a sensitivity analysis of the magnitude of graph modifications, are provided in the Appendix. The sensitivity analysis indicates that M2D can substantially reduce the distillation loss while introducing only limited changes to the original node features and graph structure.

4.3 GENERALIZATION TO MULTIPLE DOWNSTREAM ARCHITECTURES

We next investigate whether the behavior encoded in the distilled graph is preserved across downstream architectures. We evaluate whether a single distilled graph enables different models, trained independently from scratch, to maintain the teacher's behavior. Downstream models with distinct architectures are considered: GCN (Kipf & Welling, 2017), SGC (Wu et al., 2019), and MLP. Each model is trained using only the distilled graph generated with a GCN as the M2D student and training labels, without further access to the teacher. We compare the resulting models trained on the original and distilled graphs in terms of accuracy, DP, EQOP, and prediction fidelity to the teacher.

Tables 2 and 3 evaluate M2D on CORA with a GAT teacher and NBA with a FairGNN teacher, using independently trained GCN, SGC, and MLP as downstream models, respectively. Across both datasets, distillation consistently improves student fidelity to the teacher while preserving comparable predictive accuracy. On NBA, M2D also reduces DP and EQ for most student models,

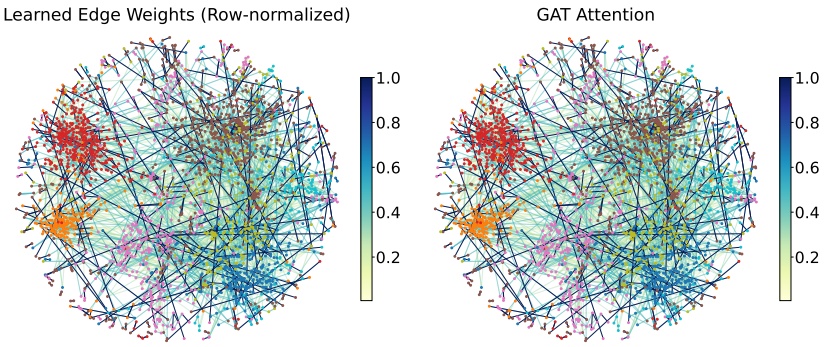

Figure 7: Visualization of row-normalized M2D learned weights (left) and layer-1 GAT attention coefficients (right) for CORA. Edge color intensity and width are proportional to the magnitude of the weight (darker and thicker edges indicate higher importance). Node colors indicate class labels. The learned weights approximate the GAT coefficients (Pearson's correlation = 0.84).

	GCN		SGC		MLP	
	Accuracy (↑)	Fidelity (↑)	Accuracy (↑)	Fidelity (↑)	Accuracy (↑)	Fidelity (↑)
Vanilla	79.23 ± 0.26	91.45 ± 0.52	78.91 ± 0.42	91.23 ± 0.61	63.76 ± 0.11	65.15 ± 0.29
$\tilde{\mathcal{G}}$(Feat.)	81.54 ± 0.41	95.21 ± 0.35	$\mathbf{80.97 \pm 0.29}$	93.54 ± 0.50	$\mathbf{65.06 \pm 0.31}$	$\mathbf{68.29 \pm 0.25}$
$\tilde{\mathcal{G}}$(Adj)	80.61 ± 0.19	94.19 ± 0.24	79.52 ± 0.41	94.01 ± 0.42	-	-
$\tilde{\mathcal{G}}$(Both)	$\mathbf{81.98 \pm 0.27}$	$\mathbf{95.98 \pm 0.52}$	80.19 ± 0.30	$\mathbf{94.60 \pm 0.13}$	-	-

Table 2: Generalization of M2D with a GAT teacher on CORA using different downstream architectures trained only on the augmented graph. We evaluate M2D with feature and edge augmentation individually and jointly across GCN, SGC, and MLP. Fidelity consistently increases with distillation, indicating improved alignment between the downstream models and teacher predictions.

demonstrating that the fairness behavior of the FairGNN teacher is preserved across downstream architectures, including those those different from the student architecture. These results further support our hypothesis that the distilled graph preserves the behavior encoded by the teacher. Results from additional datasets are provided in the Appendix.

	GCN				SGC			
	Accuracy (↑)	Fidelity (↑)	DP (↓)	EQOP (↓)	Accuracy (↑)	Fidelity (↑)	DP (↓)	EQOP (↓)
Vanilla	72.23 ± 0.42	87.21 ± 0.33	8.10 ± 0.51	22.19 ± 0.62	71.23 ± 0.16	86.84 ± 0.41	9.10 ± 0.52	22.58 ± 0.53
$\tilde{\mathcal{G}}$(Feat.)	71.15 ± 0.29	$\mathbf{94.12 \pm 0.31}$	$\mathbf{6.14 \pm 0.23}$	16.29 ± 0.23	$\mathbf{70.67 \pm 0.24}$	89.91 ± 0.53	7.04 ± 0.30	18.24 ± 0.24
$\tilde{\mathcal{G}}$(Adj)	71.59 ± 0.37	92.97 ± 0.24	6.97 ± 0.28	$\mathbf{15.02 \pm 0.42}$	70.12 ± 0.31	90.05 ± 0.24	$\mathbf{6.52 \pm 0.25}$	$\mathbf{18.01 \pm 0.62}$
$\tilde{\mathcal{G}}$(Both)	$\mathbf{71.81 \pm 0.43}$	93.05 ± 0.46	6.54 ± 0.19	17.02 ± 0.17	70.06 ± 0.28	$\mathbf{90.67 \pm 0.31}$	6.91 ± 0.29	18.06 ± 0.41

Table 3: Generalization of M2D with a FairGNN teacher on NBA using GCN and SGC as downstream architectures trained only on the augmented graph. Fidelity consistently increases with distillation, indicating improved alignment between student and teacher predictions, while fairness is generally improved, and predictive performance remains comparable.

5 CONCLUSION

We introduced Model-to-Data (M2D) distillation, a framework for transferring properties from sophisticated GNNs into graph data rather than into a particular student model. By modifying node features and graph structure, M2D enables simpler models trained with standard objectives to recover behaviors of more complex teachers, effectively trading model complexity for data complexity. Our experiments show that M2D can transfer properties from fairness-aware GNNs, Graph Attention Networks, and Graph Transformers to simpler downstream models, demonstrating that the distilled graph can serve as a reusable medium for preserving teacher behavior beyond the original model. Beyond distillation, the learned data transformations also provide insight into how to modify feature and structural information to achieve the teacher's behavior.

AI USE STATEMENT

In this work, we used generative AI tools for editing the paper. We have not used generative AI tools for assisting in the writing of proofs, provide feedback on research methodology and experiments, drafting parts of the paper, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for summarizing and analyzing existing literature. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Chirag Agarwal, Himabindu Lakkaraju, and Marinka Zitnik. Towards a unified framework for fair and stable graph representation learning. In *Uncertainty in Artificial Intelligence*, pp. 2114–2124. PMLR, 2021.

Hugo Attali, Davide Buscaldi, and Nathalie Pernelle. Rewiring techniques to mitigate oversquashing and oversmoothing in gnns: A survey. *arXiv preprint arXiv:2411.17429*, 2024.

Yu Chen, Lingfei Wu, and Mohammed Zaki. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Advances in Neural Information Processing Systems*, 33:19314–19326, 2020.

Yuzhao Chen, Yatao Bian, Xi Xiao, Yu Rong, Tingyang Xu, and Junzhou Huang. On self-distilling graph neural network. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021.

Enyan Dai and Suhang Wang. Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 680–688, 2021.

Xiang Deng and Zhongfei Zhang. Graph-free knowledge distillation for graph neural networks. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021.

Mingze Dong and Yuval Kluger. Towards understanding and reducing graph structural noise for gnns. In *International Conference on Machine Learning*, pp. 8202–8226. PMLR, 2023.

Junfeng Fang, Xinglin Li, Yongduo Sui, Yuan Gao, Guibin Zhang, Kun Wang, Xiang Wang, and Xiangnan He. Exgc: Bridging efficiency and explainability in graph condensation. In *Proceedings of the ACM Web Conference 2024*, pp. 721–732, 2024.

Bahare Fatemi, Layla El Asri, and Seyed Mehran Kazemi. Slaps: Self-supervision improves structure learning for graph neural networks. *Advances in Neural Information Processing Systems*, 34: 22667–22681, 2021.

Kaituo Feng, Changsheng Li, Ye Yuan, and Guoren Wang. Freekd: Free-direction knowledge distillation for graph neural networks. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 357–366, 2022.

Qizhang Feng, Zhimeng Stephen Jiang, Ruiquan Li, Yicheng Wang, Na Zou, Jiang Bian, and Xia Hu. Fair graph distillation. *Advances in Neural Information Processing Systems*, 36:80644–80660, 2023.

Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. Learning discrete structures for graph neural networks. In *International Conference on Machine Learning*, pp. 1972–1982. PMLR, 2019.

Xinyi Gao, Tong Chen, Yilong Zang, Wentao Zhang, Quoc Viet Hung Nguyen, Kai Zheng, and Hongzhi Yin. Graph condensation for inductive node representation learning. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pp. 3056–3069, 2024a.

Xinyi Gao, Tong Chen, Wentao Zhang, Yayong Li, Xiangguo Sun, and Hongzhi Yin. Graph condensation for open-world graph learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 851–862, 2024b.

Xinyi Gao, Hongzhi Yin, Tong Chen, Guanhua Ye, Wentao Zhang, and Bin Cui. Robgc: Towards robust graph condensation. *IEEE Transactions on Knowledge and Data Engineering*, 37(8):4791–4804, 2025.

Alberto P García-Plaza, Víctor Fresno, Raquel Martínez Unanue, and Arkaitz Zubiaga. Using fuzzy logic to leverage html markup for web page representation. *IEEE Transactions on Fuzzy Systems*, 25(4):919–933, 2016.

Johannes Gasteiger, Stefan Weißenberger, and Stephan Günnemann. Diffusion improves graph learning. *Advances in Neural Information Processing Systems*, 32, 2019.

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*, pp. 1263–1272. PMLR, 2017.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

Hongyu Guo and Yongyi Mao. Interpolating graph pair to regularize graph classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 7766–7774, 2023.

Jiongyu Guo, Defang Chen, and Can Wang. Alignahead: Online cross-layer knowledge extraction on graph neural networks. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2022.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 2017.

Mohammad Hashemi, Shengbo Gong, Juntong Ni, Wenqi Fan, B Aditya Prakash, and Wei Jin. A comprehensive survey on graph reduction: Sparsification, coarsening, and condensation. *arXiv preprint arXiv:2402.03358*, 2024.

Huarui He, Jie Wang, Zhanqiu Zhang, and Feng Wu. Compressing deep graph neural networks via adversarial knowledge distillation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 534–544, 2022.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Hans Hofmann. Statlog (German Credit Data). UCI Machine Learning Repository, 1994.

Cuiying Huo, Di Jin, Yawen Li, Dongxiao He, Yu-Bin Yang, and Lingfei Wu. T2-gnn: Graph neural networks for graphs with incomplete features and structure via teacher-student distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 4339–4346, 2023.

Jongwon Jeong, Hoyeop Lee, Hyui Geon Yoon, Beomyoung Lee, Junhee Heo, Geonsoo Kim, and Kim Jin Seon. igraphmix: Input graph mixup method for node classification. In *International Conference on Learning Representations*, 2024.

Wei Jin, Xianfeng Tang, Haoming Jiang, Zheng Li, Danqing Zhang, Jiliang Tang, and Bing Yin. Condensing graphs via one-step gradient matching. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 720–730, 2022a.

Wei Jin, Lingxiao Zhao, Shichang Zhang, Yozen Liu, Jiliang Tang, and Neil Shah. Graph condensation for graph neural networks. In *International Conference on Learning Representations*, 2022b.

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*, 35(4):4656–4667, 2022.

Vassilis Kalofolias. How to learn a graph from smooth signals. In *Artificial Intelligence and Statistics*, pp. 920–929. PMLR, 2016.

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

Kezhi Kong, Guohao Li, Mucong Ding, Zuxuan Wu, Chen Zhu, Bernard Ghanem, Gavin Taylor, and Tom Goldstein. Robust optimization as data augmentation for large-scale graphs. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 60–69. IEEE, 2022.

Kwang Hee Lee, Hyun Ji Jeong, and Myoung Ho Kim. A cold-start aware group recommendation method based on an augmented graph and knowledge distillation. *Applied Soft Computing*, 180: 113341, 2025.

Shiye Lei and Dacheng Tao. A comprehensive survey of dataset distillation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(1):17–32, 2023.

Zhixun Li, Liang Wang, Xin Sun, Yifan Luo, Yanqiao Zhu, Dingshuo Chen, Yingtao Luo, Xiangxin Zhou, Qiang Liu, Shu Wu, et al. Gslb: The graph structure learning benchmark. *Advances in Neural Information Processing Systems*, 36:30306–30318, 2023.

Yang Liu, Deyu Bo, and Chuan Shi. Graph distillation with eigenbasis matching. In *Forty-first International Conference on Machine Learning*, 2024a.

Zhanyu Liu, Chaolv Zeng, and Guanjie Zheng. Graph data condensation via self-expressive graph structure reconstruction. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1992–2002, 2024b.

Runze Mao, Wenqi Fan, and Qing Li. Gcare: Mitigating subgroup unfairness in graph condensation through adversarial regularization. *Applied Sciences*, 13(16):9166, 2023.

Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *International Conference on Learning Representations*, 2020.

Avishkar Saha, Oscar Mendez, Chris Russell, and Richard Bowden. Learning adaptive neighborhoods for graph neural networks. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22484–22493. IEEE, 2023.

Sina Sajadmanesh, Ali Shahin Shamsabadi, Aurélien Bellet, and Daniel Gatica-Perez. {GAP}: Differentially private graph neural networks with aggregation perturbation. In *32nd USENIX Security Symposium*, pp. 3223–3240, 2023.

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.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

Seiyun Shin, Ilan Shomorony, and Han Zhao. Efficient learning of linear graph neural networks via node subsampling. *Advances in Neural Information Processing Systems*, 36:55479–55501, 2023.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pp. 1067–1077, 2015.

Yijun Tian, Chuxu Zhang, Zhichun Guo, Xiangliang Zhang, and Nitesh Chawla. Learning MLPs on graphs: A unified view of effectiveness, robustness, and efficiency. In *International Conference on Learning Representations*, 2023.

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.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Can Wang, Sheng Zhou, Kang Yu, Defang Chen, Bolang Li, Yan Feng, and Chun Chen. Collaborative knowledge distillation for heterogeneous information network embedding. In *Proceedings of the ACM Web Conference*, pp. 1631–1639, 2022a.

Qianqian Wang, Junhao Zhang, Long Li, Lishan Qiao, Pew-Thian Yap, and Mingxia Liu. Graph augmentation guided federated knowledge distillation for multisite functional mri analysis. *Pattern Recognition*, 164:111559, 2025.

Xiao Wang, Hongrui Liu, Chuan Shi, and Cheng Yang. Be confident! towards trustworthy graph neural networks via confidence calibration. *Advances in Neural Information Processing Systems*, 34:23768–23779, 2021.

Yu Wang, Yuying Zhao, Yushun Dong, Huiyuan Chen, Jundong Li, and Tyler Derr. Improving fairness in graph neural networks via mitigating sensitive attribute leakage. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1938–1948, 2022b.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International Conference on Machine Learning*, pp. 6861–6871. Pmlr, 2019.

Lirong Wu, Haitao Lin, Yufei Huang, and Stan Z Li. Knowledge distillation improves graph structure augmentation for graph neural networks. *Advances in Neural Information Processing Systems*, 35:11815–11827, 2022.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

Zhe Xu, Yuzhong Chen, Menghai Pan, Huiyuan Chen, Mahashweta Das, Hao Yang, and Hanghang Tong. Kernel ridge regression-based graph dataset distillation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2850–2861, 2023.

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*, pp. 1848–1856, 2020.

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*, pp. 1227–1237, 2021.

Chenxiao Yang, Qitian Wu, and Junchi Yan. Geometric knowledge distillation: Topology compression for graph neural networks. *Advances in Neural Information Processing Systems*, 35: 29761–29775, 2022.

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*, pp. 7074–7083, 2020.

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.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33:5812–5823, 2020.

Chunhai Zhang, Jie Liu, Kai Dang, and Wenzheng Zhang. Multi-scale distillation from multiple graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 4337–4344, 2022a.

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*, 2022b.

Tianle Zhang, Yuchen Zhang, Kun Wang, Kai Wang, Beining Yang, Kaipeng Zhang, Wenqi Shao, Ping Liu, Joey Tianyi Zhou, and Yang You. Two trades is not baffled: Condensing graph via crafting rational gradient matching. *arXiv preprint arXiv:2402.04924*, 2024.

Wentao Zhang, Yuezihan Jiang, Yang Li, Zeang Sheng, Yu Shen, Xupeng Miao, Liang Wang, Zhi Yang, and Bin Cui. Rod: reception-aware online distillation for sparse graphs. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 2232–2242, 2021.

Tong Zhao, Xianfeng Tang, Danqing Zhang, Haoming Jiang, Nikhil Rao, Yiwei Song, Pallav Agrawal, Karthik Subbian, Bing Yin, and Meng Jiang. Autogda: Automated graph data augmentation for node classification. In *Learning on Graphs Conference*. PMLR, 2022.

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*, 2022.

Xin Zheng, Miao Zhang, Chunyang Chen, Quoc Viet Hung Nguyen, Xingquan Zhu, and Shirui Pan. Structure-free graph condensation: From large-scale graphs to condensed graph-free data. *Advances in Neural Information Processing Systems*, 36:6026–6047, 2023.

Zhiyao Zhou, Sheng Zhou, Bochao Mao, Xuanyi Zhou, Jiawei Chen, Qiaoyu Tan, Daochen Zha, Yan Feng, Chun Chen, and Can Wang. Opengsl: A comprehensive benchmark for graph structure learning. *Advances in Neural Information Processing Systems*, 36:17904–17928, 2023.

Yanqiao Zhu, Weizhi Xu, Jinghao Zhang, Yuanqi Du, Jieyu Zhang, Qiang Liu, Carl Yang, and Shu Wu. A survey on graph structure learning: Progress and opportunities. *arXiv preprint arXiv:2103.03036*, 2021.

Yuanxin Zhuang, Lingjuan Lyu, Chuan Shi, Carl Yang, and Lichao Sun. Data-free adversarial knowledge distillation for graph neural networks. *arXiv preprint arXiv:2205.03811*, 2022.

Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2847–2856, 2018.

A TECHNICAL APPENDICES AND SUPPLEMENTARY MATERIAL

This section contains the following:

- Background and evaluation metrics
- Dataset and baseline descriptions
- Proofs of Theorems 1 and 2
- M2D training algorithm and experimental setup
- Additional results and visualizations
- Sensitivity and downstream architecture generalization analyses
- Computational complexity and reproducibility details

GRAPH NEURAL NETWORKS

We adopt GCN and SGC as the students and GAT as the teacher in several experiments Kipf & Welling (2017); Veličković et al. (2018); Wu et al. (2019). All models belong to the class of message-passing neural networks, which learn node representations $\mathbf{h}_v$ used for prediction $y_v = f(\mathbf{h}_v)$. These representations are computed via iterative AGGREGATE and COMBINE operations:

$$\mathbf{h}_v^{t+1} = \mathsf{COMBINE}^t\Big(\mathbf{h}_v^t,\ \mathsf{AGGREGATE}^t\big(\{\mathbf{h}_u^t : A_{uv} = 1\}\big)\Big),$$

where $\mathbf{h}_v^t$ denotes the embedding of the node v at layer t.

The primary difference between GCN and GAT lies in the aggregation mechanism. GCN employs fixed, normalized adjacency weights to average neighbor features, whereas GAT learns attention coefficients that assign different importance to neighbors, enabling adaptive and data-dependent message passing.

For GCN, the layer-wise update is given by:

$$\mathbf{h}_v^{t+1} = \sigma\big(\frac{1}{\sqrt{|N(v)|+1}}\mathbf{W}^t\big(\frac{\mathbf{h}_v^t}{\sqrt{|N(v)|+1}} + \sum_{u\in N(v)}\frac{\mathbf{h}_u^t}{\sqrt{|N(u)|+1}}\big)\big),$$

For GAT, the update rule is:

$$\mathbf{h}_v^{t+1} = \sigma\big(\sum_{u\in N(v)\cup\{v\}} \alpha_{vu}\,\mathbf{W}\mathbf{h}_u^t\big),$$

where α_{vu} denotes the attention coefficient on edge (v, u):

$$\alpha_{vu} = \frac{\exp\Big(\mathrm{LeakyReLU}\big(\mathbf{a}^\top[\mathbf{h}_v^t\mathbf{W},\ \mathbf{h}_u^t\mathbf{W}]\big)\Big)}{\sum_{k\in N(v)\cup\{v\}}\exp\Big(\mathrm{LeakyReLU}\big(\mathbf{a}^\top[\mathbf{h}_v^t\mathbf{W},\ \mathbf{h}_k^t\mathbf{W}]\big)\Big)},$$

with learnable parameters $\mathbf{W}$ and $\mathbf{a}$.

SGC simplifies the GCN architecture by removing intermediate nonlinearities and collapsing multiple propagation steps into a single linear transformation. Given the normalized adjacency matrix $\hat{\mathbf{A}}$, the node representations after K propagation steps are computed as:

$$\mathbf{H}^{(K)} = \hat{\mathbf{A}}^K\mathbf{X},$$

where $\mathbf{X}$ denotes the input node features. The final node representations are then passed through a linear classifier:

$$\mathbf{Y} = \mathbf{H}^{(K)}\mathbf{W},$$

where $\mathbf{W}$ is a learnable weight matrix. By removing intermediate nonlinear transformations, SGC provides a simpler downstream architecture while retaining multi-hop neighborhood information through repeated graph propagation.

We also consider a **Graph Transformer (GT)** as a teacher model. More specifically, we consider Graphormer Ying et al. (2021) as a representative GT. GTs are built by stacking multiple Transformer layers Vaswani et al. (2017), where each layer comprises a self-attention module followed by a feed-forward network.

Let $\mathbf{H} \in \mathbb{R}^{n \times d}$ denote the input to the self-attention module. This input is linearly projected into query, key, and value representations via learnable matrices $\mathbf{W}_Q$, $\mathbf{W}_K$, and $\mathbf{W}_V$:

$$\mathbf{Q} = \mathbf{H}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{H}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{H}\mathbf{W}_V.$$

The self-attention operation is then computed as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\big(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_K}}\big)\mathbf{V},$$

where d_K denotes the dimensionality of the key vectors.

Evaluation Metrics

Utility: We use accuracy to evaluate utility.

Fairness: We measure group fairness with respect to a sensitive attribute s, where $a = 1$ denotes the sensitive group and $a = 0$ the non-sensitive group. Lower values indicate better fairness.

Demographic Parity (DP) measures the difference in positive prediction rates across groups:

$$DP(\hat{Y}, a) = \big|p(\hat{Y} = 1 \mid a = 1) - p(\hat{Y} = 1 \mid a = 0)\big|. \tag{6}$$

Equality of Opportunity (EQOP) measures the difference in true positive rates:

$$EQOP(\hat{Y}, a, y) = \big|p(\hat{Y} = 1 \mid a = 1, y = 1) - p(\hat{Y} = 1 \mid a = 0, y = 1)\big|. \tag{7}$$

DP and EQOP both measure bias. We can get corresponding fairness by subtracting them from 1.

Dataset Details

Dataset for Fairness Evaluation:

Simulation: We generate a synthetic graph with 1000 nodes using a stochastic block model with two communities of sizes 600 (majority) and 400 (minority), and set intra and inter-block edge probabilities to achieve high assortativity (≈ 0.77). The sensitive attribute a is generated conditionally on the label $\mathbf{y}$: $a \sim \text{Bernoulli}(p)$ if $y = 1$ and $a \sim \text{Bernoulli}(1 - p)$ otherwise, we set $p = 0.7$ to induce bias, creating high correlation between a and $\mathbf{y}$. This setup introduces structural bias, as nodes are more likely to connect within the same community, leading to label–group correlations in the graph topology. Each node has 20 features, of which 8 are noisy, sampled from $\mathcal{N}(0, 1)$, while the remaining features are generated as $\gamma \cdot y + \mathcal{N}(0, 1)$. This introduces feature-level bias, as informative attributes are correlated with the label and, indirectly, with the sensitive attribute, reinforcing disparities present in both structure and features.

German (Hofmann, 1994): Vertices represent 1000 individuals with 20 financial and demographic features. Edges are computed from vertex similarity. The prediction task is to classify credit (good vs. bad credit), with gender treated as the sensitive attribute.

NBA (Dai & Wang, 2021): This dataset consists of 403 NBA players connected via their Twitter interactions. Node features capture player statistics, and the task is to predict whether a player's salary is above or below the median. Nationality is treated as a sensitive attribute.

Dataset for Improved Performance Evaluation

Statistics for the datasets used to analyze the performance of GAT and GT are provided in Table 4.

BASELINES

Vanilla: The vanilla model is the student model without any distillation.

FairGNN Dai & Wang (2021): Jointly trains a GNN classifier with a neural network adversary, where the adversary encourages the learned representations to remove sensitive information and thus mitigate bias.

FairVGNN Wang et al. (2022b): FairVGNN learns fair feature views by identifying and masking features correlated with sensitive attributes, explicitly accounting for shifts in these correlations induced by feature propagation. Conditioned on the resulting views, it adaptively regularizes the encoder parameters to suppress the influence of sensitive-related information.

NIFTY Agarwal et al. (2021): NIFTY ensures fairness and stability by combining a joint fairness–stability objective with Lipschitz-based layer-wise weight normalization. It provides theoretical guarantees for counterfactual fairness and stable representations, and achieves strong empirical performance on high-stakes datasets.

GT Ying et al. (2021): Transformer-based model for graph representation learning that achieves strong performance across a wide range of tasks. Its core design centers on explicitly encoding graph structure into the Transformer architecture through centrality encoding, edge encoding, and spatial encoding, enabling effective modeling of node relationships beyond standard message passing.

Dataset	Type	Nodes	Edges	Classes	Features
CORA	Homophilic	2708	5429	7	1433
CITESEER	Homophilic	3327	4732	6	3703
PHOTO	Homophilic	7650	238163	8	745
CORNELL	Heterophilic	183	298	5	1703
TEXAS	Heterophilic	183	325	5	1703

Table 4: Dataset statistics Sen et al. (2008); Shchur et al. (2018); García-Plaza et al. (2016)

PROOF OF THEOREMS

Our theorems are based on feature augmentations. A similar analysis can be provided for structural augmentations where the learned structure approximates the representations from the teacher model.

THEOREM 1

Proof. The demographic parity of the student f_s is

$$\mathrm{DP}(f_s) = \big|\mathbb{E}[\hat{Y}_s \mid a=0] - \mathbb{E}[\hat{Y}_s \mid a=1]\big|.$$

Adding and subtracting the corresponding teacher group means gives

$$\begin{aligned}\mathrm{DP}(f_s) =& \big|\mathbb{E}[\hat{Y}_s \mid a=0] - \mathbb{E}[\hat{Y}_T \mid a=0] \\ &+ \mathbb{E}[\hat{Y}_T \mid a=0] - \mathbb{E}[\hat{Y}_T \mid a=1] \\ &+ \mathbb{E}[\hat{Y}_T \mid a=1] - \mathbb{E}[\hat{Y}_s \mid a=1]\big|.\end{aligned}$$

Applying the triangle inequality,

$$\begin{aligned}\mathrm{DP}(f_s) \le& \mathrm{DP}(f_T) \\ &+ \big|\mathbb{E}[\hat{Y}_s - \hat{Y}_T \mid a=0]\big| \\ &+ \big|\mathbb{E}[\hat{Y}_s - \hat{Y}_T \mid a=1]\big|.\end{aligned}$$

For each group $g \in \{0,1\}$,

$$\Big|\mathbb{E}[\hat{Y}_s - \hat{Y}_T \mid a=g]\Big| \le \mathbb{E}\Big[|\hat{Y}_s - \hat{Y}_T|\Big|a=g\Big].$$

For Bernoulli distributions, the total-variation distance satisfies

$$D_{\mathrm{TV}}\big(\mathrm{Bern}(\hat{Y}_T), \mathrm{Bern}(\hat{Y}_s)\big) = |\hat{Y}_T - \hat{Y}_s|.$$

By Pinsker's inequality,

$$|\hat{Y}_T - \hat{Y}_s| \le \sqrt{\frac{1}{2} D_{\mathrm{KL}}\big(\mathrm{Bern}(\hat{Y}_T), \mathrm{Bern}(\hat{Y}_s)\big)}.$$

Taking the conditional expectation over group g gives

$$\mathbb{E}\big[|\hat{Y}_T - \hat{Y}_s| \mid a = g\big] \le \mathbb{E}\big[\sqrt{\frac{1}{2} D_{\mathrm{KL}}\big(\mathrm{Bern}(\hat{Y}_T), \mathrm{Bern}(\hat{Y}_s)\big)} \mid a = g\big].$$

Since the square-root function is concave, Jensen's inequality yields

$$\begin{aligned}\mathbb{E}\Big[|\hat{Y}_T - \hat{Y}_s| \Big| a = g\Big] &\le \sqrt{\frac{1}{2}\mathbb{E}\big[D_{\mathrm{KL}}\big(\mathrm{Bern}(\hat{Y}_T), \mathrm{Bern}(\hat{Y}_s)\big) \mid a = g\big]} \\ &= \sqrt{\frac{\kappa_g}{2}}.\end{aligned}$$

where, $\kappa_g = \mathbb{E}\big[D_{\mathrm{KL}}\big(\mathrm{Bern}(\hat{Y}_T), \mathrm{Bern}(\hat{Y}_s)\big) \big| a = g\big]$

Therefore,

$$\mathrm{DP}(f_s) \le \mathrm{DP}(f_T) + \sqrt{\frac{\kappa_0}{2}} + \sqrt{\frac{\kappa_1}{2}}.$$

This bound depends only on the final teacher-student prediction discrepancy. We next characterize the variation between the M2D student f_s and a newly trained downstream model $f_{s'}$ on the frozen graph $\tilde{\mathcal{G}}$. Conditioned on an input $\tilde{\mathcal{G}}$, let $\eta = \Pr(Y \mid \tilde{\mathcal{G}})$ be the Bayes optimal conditional probability of class label Y.

Applying Pinsker's inequality pointwise gives

$$|\hat{Y}_{s'} - \eta| \le \sqrt{\frac{1}{2} D_{\mathrm{KL}}\big(\mathrm{Bern}(\eta), \mathrm{Bern}(\hat{Y}_{s'})\big)}.$$

Taking the conditional expectation and applying Jensen's inequality,

$$\begin{aligned}\mathbb{E}\Big[|\hat{Y}_{s'} - \eta| \Big| a = g\Big] &\le \sqrt{\frac{1}{2}\mathbb{E}\big[D_{\mathrm{KL}}\big(\mathrm{Bern}(\eta), \mathrm{Bern}(\hat{Y}_{s'})\big) \big| a = g\big]} \\ &= \sqrt{\frac{\xi_g(f_{s'})}{2}}.\end{aligned}$$

Now consider f_s and $f_{s'}$.

$$|\hat{Y}_s - \hat{Y}_{s'}| = |(\hat{Y}_s - \eta) + (\eta - \hat{Y}_{s'})| \le |\hat{Y}_s - \eta| + |\eta - \hat{Y}_{s'}| = |\hat{Y}_s - \eta| + |\hat{Y}_{s'} - \eta|.$$

Taking the conditional expectation within group g,

$$\begin{aligned}\mathbb{E}\Big[|\hat{Y}_s - \hat{Y}_{s'}| \Big| a = g\Big] &\le \mathbb{E}\Big[|\hat{Y}_s - \eta| \Big| a = g\Big] + \mathbb{E}\Big[|\hat{Y}_{s'} - \eta| \Big| a = g\Big] \\ &\le \sqrt{\frac{\xi_g(f_s)}{2}} + \sqrt{\frac{\xi_g(f_{s'})}{2}}.\end{aligned}$$

Similarly,

$$|\hat{Y}_{s'} - \hat{Y}_T| = |(\hat{Y}_{s'} - \eta) + (\eta - \hat{Y}_s) + (\hat{Y}_s - \hat{Y}_T)| \le |\hat{Y}_{s'} - \eta| + |\hat{Y}_s - \eta| + |\hat{Y}_s - \hat{Y}_T|.$$

Therefore,

$$\mathbb{E}\Big[|\hat{Y}_{s'} - \hat{Y}_T| \Big| a = g\Big] \le \sqrt{\frac{\xi_g(f_s)}{2}} + \sqrt{\frac{\xi_g(f_{s'})}{2}} + \sqrt{\frac{\kappa_g}{2}}.$$

Applying the same demographic-parity decomposition used above to the downstream student $f_{s'}$ yields

$$\begin{aligned}\mathrm{DP}(f_{s'}) \le \mathrm{DP}(f_T) \\ + \mathbb{E}\big[|\hat{Y}_{s'} - \hat{Y}_T| \mid a = 0\big] \\ + \mathbb{E}\big[|\hat{Y}_{s'} - \hat{Y}_T| \mid a = 1\big].\end{aligned}$$

Substituting the previous bound for each sensitive group gives

$$\mathrm{DP}(f_{s'}) \le \mathrm{DP}(f_T) + \sum_{g\in\{0,1\}} \left(\sqrt{\frac{\kappa_g}{2}} + \left(\sqrt{\frac{\xi_g(f_s)}{2}} + \sqrt{\frac{\xi_g(f_{s'})}{2}}\right)\right).$$

Therefore,

$$\mathrm{DP}(f_{s'}) - \mathrm{DP}(f_T) \le \sum_{g\in\{0,1\}} \left(\sqrt{\frac{\kappa_g}{2}} + \left(\sqrt{\frac{\xi_g(f_s)}{2}} + \sqrt{\frac{\xi_g(f_{s'})}{2}}\right)\right).$$

Similarly, we can get

$$\mathrm{DP}(f_T) - \mathrm{DP}(f_{s'}) \le \sum_{g\in\{0,1\}} \left(\sqrt{\frac{\kappa_g}{2}} + \left(\sqrt{\frac{\xi_g(f_s)}{2}} + \sqrt{\frac{\xi_g(f_{s'})}{2}}\right)\right).$$

Hence

$$|\mathrm{DP}(f_T) - \mathrm{DP}(f_{s'})| \le \sum_{g\in\{0,1\}} \left(\sqrt{\frac{\kappa_g}{2}} + \left(\sqrt{\frac{\xi_g(f_s)}{2}} + \sqrt{\frac{\xi_g(f_{s'})}{2}}\right)\right).$$

Therefore, the fairness difference between a new downstream model and the teacher is controlled by the distillation error and how well the original student and the new downstream model approximate the optimal predictor on the transformed graph, while both the student and the downstream model optimize cross-entropy loss. □

THEOREM 2

Proof. Consider a single-layer GAT with teacher representation

$$h_{T,i} = \sigma\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij} W x_j\right).$$

Using a first-order Taylor approximation, there exist a linear map $\widetilde{W}$ and residual b_i such that

$$h_{T,i} = \sum_{j\in\mathcal{N}(i)} \alpha_{ij} \widetilde{W} x_j + b_i.$$

Let the teacher and student logits be

$$z_{T,i} = U_T h_{T,i}, \qquad z_{s,i} = U_s h_{s,i},$$

and define their temperature-scaled predictive distributions as

$$p_{T,i} = \mathrm{softmax}(z_{T,i}/\tau), \qquad p_{s,i} = \mathrm{softmax}(z_{s,i}/\tau).$$

Let

$$R = \begin{bmatrix} 1 & 0 & \cdots & 0 & -1 \\ 0 & 1 & \cdots & 0 & -1 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & -1 \end{bmatrix} \in \mathbb{R}^{(C-1)\times C},$$

so that, for any logit vector $\mathbf{z} \in \mathbb{R}^C$,

$$R_{\mathbf{z}} = [z_1 - z_C, \ldots, z_{C-1} - z_C]^\top.$$

Since, for any logit vector $\mathbf{z}$ with $\mathbf{p} = \mathrm{softmax}(\mathbf{z}/\tau)$,

$$R_{\mathbf{z}} = \tau \begin{bmatrix} \log(p_1/p_C) \\ \vdots \\ \log(p_{C-1}/p_C) \end{bmatrix},$$

Since R subtracts the logit of the reference class C from the remaining class logits,

$$R_{\mathbf{z}_{s,i}} = \begin{bmatrix} z_{s,i,1} - z_{s,i,C} \\ \vdots \\ z_{s,i,C-1} - z_{s,i,C} \end{bmatrix},$$

and similarly for the teacher.

Therefore,

$$z_{s,i,c} - z_{s,i,C} = \tau \log \left(\frac{p_{s,i,c}}{p_{s,i,C}} \right).$$

Likewise,

$$z_{T,i,c} - z_{T,i,C} = \tau \log \left(\frac{p_{T,i,c}}{p_{T,i,C}} \right).$$

We define the contrast-logit discrepancy as:

$$\boldsymbol{\epsilon}_i = R_{\mathbf{z}_{s,i}} - R_{\mathbf{z}_{T,i}}.$$

Therefore,

$$RU_s \mathbf{h}_{s,i} = RU_T \mathbf{h}_{T,i} + \boldsymbol{\epsilon}_i.$$

Let

$$\kappa_i = D_{\mathrm{KL}}(p_{T,i} \ , \ p_{s,i}).$$

By Pinsker's inequality,

$$\|p_{T,i} - p_{s,i}\|_1 \leq \sqrt{2\kappa_i}.$$

Hence, as $\kappa_i \to 0$,

$$p_{s,i} \to p_{T,i}.$$

Since finite logits imply strictly positive softmax probabilities, $p_{T,i,c} > 0$ for all classes c, each function

$$\log \left(\frac{p_c}{p_C} \right)$$

is continuous at $\mathbf{p}_{T,i}$. Therefore, as $\mathbf{p}_{s,i} \to \mathbf{p}_{T,i}$,

$$\log \left(\frac{p_{s,i,c}}{p_{s,i,C}} \right) - \log \left(\frac{p_{T,i,c}}{p_{T,i,C}} \right) \to 0$$

for every $c = 1, \ldots, C-1$. Hence,

$$\|\boldsymbol{\epsilon}_i\|_2 \to 0.$$

Define

$$\bar{U}_s = RU_s, \qquad \bar{U}_T = RU_T.$$

Then

$$\bar{U}_s h_{s,i} = \bar{U}_T h_{T,i} + \epsilon_i.$$

Since multiple student representations can produce the same output through a linear prediction head, we adopt the minimum-ℓ_2-norm solution as a canonical choice, consistent with the implicit bias of zero-initialized gradient descent in underdetermined linear least-squares problems. Therefore,

$$h_{s,i} = \bar{U}_s^\dagger \left(\bar{U}_T h_{T,i} + \epsilon_i \right).$$

Substituting the teacher representation gives

$$h_{s,i} = \bar{U}_s^\dagger \left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij} \bar{U}_T \widetilde{W} x_j + \bar{U}_T b_i + \epsilon_i \right).$$

Let

$$Q = \bar{U}_s^\dagger \bar{U}_T \widetilde{W}$$

and

$$r_i = \bar{U}_s^\dagger \left(\bar{U}_T b_i + \epsilon_i \right).$$

Then

$$h_{s,i} = \sum_{j \in \mathcal{N}(i)} \alpha_{ij} Q x_j + r_i.$$

Defining

$$P_i = \sum_{j \in \mathcal{N}(i)} \alpha_{ij} Q x_j + r_i,$$

the feature learner yields

$$m_i = \sigma(f_\phi(P_i)).$$

Similarly, for an edge (i, j),

$$h_{s,i} + h_{s,j} = P_i + P_j, \qquad |h_{s,i} - h_{s,j}| = |P_i - P_j|.$$

Thus,

$$w_{ij} = \sigma \left(f_a(P_i + P_j, |P_i - P_j|) \right).$$

The residual can be decomposed as

$$\mathbf{r}_i = \bar{U}_s^\dagger \bar{U}_T \mathbf{b}_i + \bar{U}_s^\dagger \boldsymbol{\epsilon}_i.$$

Thus,

$$\|\mathbf{r}_i\|_2 \leq \|\bar{U}_s^\dagger \bar{U}_T \mathbf{b}_i\|_2 + \|\bar{U}_s^\dagger \boldsymbol{\epsilon}_i\|_2.$$

The first term arises from the local linearization of the teacher representation, whereas the second term is induced by the teacher-student prediction discrepancy. As $D_{\mathrm{KL}}(\mathbf{p}_{T,i} \, , \, \mathbf{p}_{s,i}) \to 0$, we have $\|\boldsymbol{\epsilon}_i\|_2 \to 0$.

Thus, both the learned feature masks and edge weights are measurable transformations of the teacher's attention-weighted neighborhood aggregations, up to residuals arising from local linearization and imperfect distillation. □

TRAINING ALGORITHM

Algorithm 1 shows the training of M2D. M2D performs iterative distillation by jointly refining the student model and the underlying graph structure. At each iteration, the current student representation is used to generate an augmented graph. The student is then updated using this augmented input to match the teacher outputs. Subsequently, the graph and feature learners are updated based on the refined student representation. This alternating process enables mutual improvement between representation learning and structure adaptation. The final augmented graph and features capture the teacher's behavior more effectively.

SOFTWARE AND HARDWARE

- Operating System: Linux (Red Hat Enterprise Linux 8.9 (Ootpa))
- GPU: NVIDIA A40
- Software: Python 3.8.10, torch 2.2.1

Algorithm 1 M2D Distillation

Require: Graph $\mathbf{A}$, features $\mathbf{X}$, labels $\mathbf{y}$, teacher outputs $\hat{\mathbf{y}}_T$
Initialize Θ_s^0, Θ_g^0 and $\mathbf{H}_s^0 \to f_s(\mathbf{X}, \mathbf{A})$
for $t = 1$ to $t_{\max}$ **do**
  (1) Construct augmented graph and features
  $\mathbf{M}^t \to f_\phi(\mathbf{H}_s^{t-1}), \quad \tilde{\mathbf{X}}^t \to [\mathbf{X} \odot \mathbf{M}^t]$
  $\mathbf{W}^t \to f_a(\mathbf{H}_s^{t-1})$
  $\tilde{\mathbf{A}}^t \to (1-\gamma)\mathbf{A} + \gamma\big((1-\beta)\mathbf{W}^t + \beta\, s(\mathbf{X})\big)$
  (2) Update student (inner step)
  $\mathbf{H}_s^t \to f_s(\tilde{\mathbf{X}}^t, \tilde{\mathbf{A}}^t)$
  $\hat{\mathbf{y}}_s^t \to f_c(\mathbf{H}_s^t)$
  Update Θ_s^t using $\mathcal{L}_{\text{cls}}$
  (3) Update graph and feature learner (outer step)
  $\mathbf{M}^t \to f_\phi(\mathbf{H}_s^t), \quad \mathbf{W}^t \to f_a(\mathbf{H}_s^t)$
  Update Θ_g^t using $\mathcal{L}_{\text{dis}} + \mathcal{L}_{\text{feat}} + \mathcal{L}_{\text{graph}}$
end for
return $\tilde{\mathbf{A}}^t, \tilde{\mathbf{X}}^t$

Experimental Setup

All experiments use two-dimensional learned features for all datasets, except NBA, where a single feature is learned. All GNN models consist of two layers. We use a linear prediction head for both student and teacher in all experiments. For Graph Transformer experiments, we adopt Graphormer Ying et al. (2021). All reported results are averaged over 5 independent runs with different random seeds. Hyperparameters are selected via grid search based on validation performance. To mitigate overfitting, we employ early stopping after 150 iterations. Each dataset is divided into 30% for training, 20% for validation, and 50% for testing. We use the Adam optimizer to train all model parameters.

		Teacher	MLP (Van.)	MLP (w/o)	MLP (Feat.)
Cora	GAT	83.24 ± 0.28	63.76 ± 0.11	63.85 ± 0.21	64.58 ± 0.13
	GT	77.42 ± 0.12		63.49 ± 0.18	62.79 ± 0.24
Citeseer	GAT	73.21 ± 0.21	68.71 ± 0.14	67.98 ± 0.13	69.48 ± 0.21
	GT	65.28 ± 0.11		66.96 ± 0.22	66.11 ± 0.14
Photo	GAT	91.35 ± 0.32	85.03 ± 0.09	84.18 ± 0.17	86.94 ± 0.14
	GT	94.20 ± 0.16		84.95 ± 0.16	88.47 ± 0.26
Cornell	GAT	53.67 ± 0.41	66.34 ± 0.18	59.31 ± 0.22	62.98 ± 0.15
	GT	70.59 ± 0.12		64.19 ± 0.16	68.80 ± 0.22
Texas	GAT	64.31 ± 0.24	75.27 ± 0.20	70.20 ± 0.23	71.11 ± 0.14
	GT	73.89 ± 0.14		72.15 ± 0.11	74.05 ± 0.19

Table 5: Node classification accuracy comparing teacher models (GAT, GT) and student MLP under different settings. 'Van.' denotes the student trained without distillation, 'w/o' standard distillation, and 'Feat.' M2D feature distillation. M2D improves student performance on Cora, Citeseer, and Photo. On Cornell and Texas, where the teacher underperforms the vanilla MLP, M2D degrades performance by transferring suboptimal knowledge.

Additional Results and Visualization

Table 5 compares node classification accuracy of teacher models (GAT, GT) and student MLP under different distillation settings. On homophilic datasets such as Cora, Citeseer, and Photo, M2D consistently improves student performance over both vanilla training and standard distillation, demonstrating effective knowledge transfer from the teacher. However, on heterophilic datasets such as Cornell and Texas, the vanilla MLP already outperforms the teacher models. In these cases, M2D leads to performance degradation, as the student is encouraged to mimic suboptimal teacher

	Teacher	GCN(Feat.)	GCN(Adj.)	GCN(Both)	GCN(w/0)
SIMULATION	FairGNN	94.12 ± 0.13	91.19 ± 0.21	93.25 ± 0.19	89.14 ± 0.11
	FairVGNN	96.03 ± 0.12	93.12 ± 0.26	96.91 ± 0.13	92.13 ± 0.30
	NIFTY	94.31 ± 0.12	95.12 ± 0.30	95.59 ± 0.24	92.15 ± 0.21
NBA	FairGNN	95.13 ± 0.11	92.31 ± 0.25	93.91 ± 0.24	93.63 ± 0.27
	FairVGNN	94.21 ± 0.15	90.63 ± 0.34	93.01 ± 0.13	92.21 ± 0.19
	NIFTY	96.13 ± 0.12	91.79 ± 0.15	94.59 ± 0.28	94.14 ± 0.25
GERMAN	FairGNN	96.29 ± 0.21	95.97 ± 0.14	96.25 ± 0.17	92.12 ± 0.25
	FairVGNN	94.98 ± 0.12	95.23 ± 0.34	95.72 ± 0.31	91.79 ± 0.31
	NIFTY	95.31 ± 0.27	95.27 ± 0.19	95.81 ± 0.22	91.92 ± 0.24

Table 6: Distillation accuracy of different M2D variants for fair teacher

	Teacher	GCN(Feat.)	GCN(Adj.)	GCN(Both)	GCN(w/o)	MLP(Feat.)	MLP(w/o)
CORA	GAT	95.94 ± 0.12	91.05 ± 0.09	96.12 ± 0.14	90.56 ± 0.10	68.12 ± 0.13	63.58 ± 0.22
	GT	93.08 ± 0.14	91.15 ± 0.12	93.14 ± 0.12	89.15 ± 0.12	79.12 ± 0.08	75.24 ± 0.13
CITESEER	GAT	94.14 ± 0.15	95.12 ± 0.11	97.56 ± 0.12	92.41 ± 0.13	80.16 ± 0.15	78.23 ± 0.14
	GT	94.12 ± 0.13	95.31 ± 0.12	93.12 ± 0.15	91.57 ± 0.07	83.12 ± 0.14	82.85 ± 0.18
PHOTO	GAT	96.13 ± 0.05	94.14 ± 0.18	93.89 ± 0.11	90.91 ± 0.21	76.12 ± 0.20	73.15 ± 0.31
	GT	95.18 ± 0.14	95.54 ± 0.15	93.12 ± 0.20	89.51 ± 0.11	84.12 ± 0.13	80.12 ± 0.23
CORNELL	GAT	95.12 ± 0.13	95.04 ± 0.12	92.56 ± 0.12	90.80 ± 0.13	74.13 ± 0.21	71.02 ± 0.28
	GT	80.41 ± 0.11	75.13 ± 0.15	73.14 ± 0.17	73.01 ± 0.12	86.14 ± 0.12	84.12 ± 0.21
TEXAS	GAT	97.12 ± 0.16	95.35 ± 0.13	96.10 ± 0.14	94.30 ± 0.19	79.12 ± 0.13	77.21 ± 0.25
	GT	86.13 ± 0.14	79.12 ± 0.14	74.21 ± 0.13	69.13 ± 0.14	90.63 ± 0.21	87.12 ± 0.19

Table 7: Distillation accuracy of different M2D variants for the GAT and GT teacher

representations. This highlights that the effectiveness of distillation depends on the quality of the teacher and that blindly transferring knowledge can be detrimental when the teacher is weaker than the student.

Table 6 reports the distillation accuracy of different M2D variants with fairness-aware GNN teachers. Compared with the vanilla GCN without augmentation, M2D consistently improves the student's ability to approximate the teacher. Joint feature and adjacency augmentation achieves the best or near-best performance in most settings, while feature augmentation alone remains competitive and frequently outperforms adjacency-only augmentation. These results demonstrate that M2D effectively transfers the behavior of fairness-aware teachers to simpler student architectures.

Table 7 evaluates distillation across different M2D variants, teacher architectures, and datasets. On the homophilic datasets (CORA, CITESEER, and PHOTO), GCN students generally achieve high distillation accuracy when learning from both GAT and GT teachers, whereas MLP students perform substantially worse. In contrast, on the heterophilic datasets (CORNELL and TEXAS), MLP students achieve higher distillation accuracy in several settings, particularly when distilling from GT, while GCN performance degrades in some cases. These results indicate that the effectiveness of distillation depends on the compatibility between the student architecture and the structural properties of the underlying graph.

Figure 8 compares edge importance from M2D and GAT on the heterophilic CORNELL graph. While GAT directly learns attention coefficients, M2D learns edge weights that we row-normalize for comparison, enabling a probabilistic interpretation. The visualization reveals clear differences in how edges are weighted across the two methods. Due to heterophily, the normalized M2D weights do not match GAT attention exactly but instead reflect the adjustments required for the student to approximate the teacher's behavior. This highlights the role of edge reweighting in aligning the student with attention-based representations. The difference in scale between the two plots is due to the hyperparameter γ. For the CORNELL dataset, a higher value of the hyperparameter (0.45) generates the best result.

We further analyze the effect of α and λ_{sparse} on the magnitude of changes introduced by M2D. Figures 9 and 10 report the KL divergence between student and teacher predictions together with the average Frobenius distance between the original and distilled features and adjacency matrices, respectively. Across datasets, M2D achieves substantial reductions in KL divergence with only small average changes to both features and structure. As the regularization is relaxed, larger modifications initially improve distillation, but the benefit saturates, with further changes yielding little additional

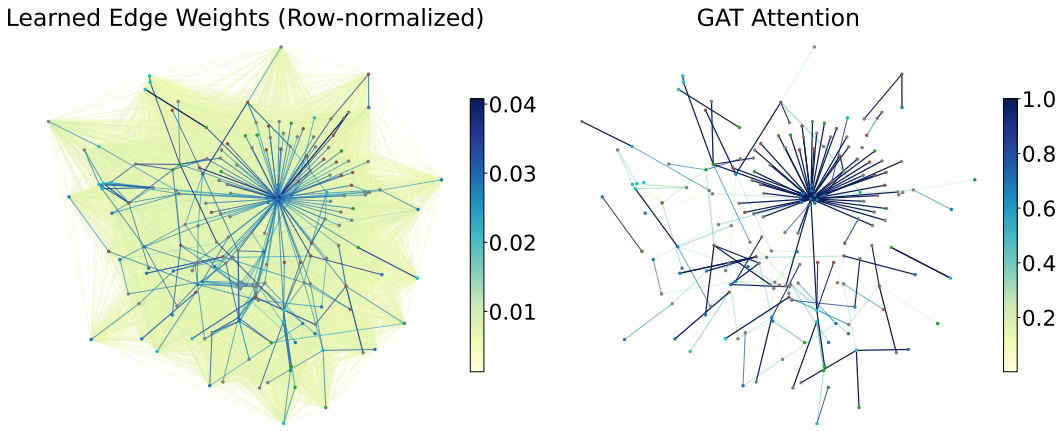

Figure 8: Visualization of edge importance on CORNELL (heterophilic) using row-normalized M2D edge weights (left) and layer-1 GAT attention coefficients (right). Edge color and width indicate magnitude, with darker and thicker edges denoting higher importance. Learned weights are row-normalized (sum to one), enabling direct probabilistic comparison with attention. Both panels share an identical layout. The left panel highlights how M2D reweights edges to approximate GAT behavior. However, the Pearson correlation coefficient between the edge weights is 0.11 (low).

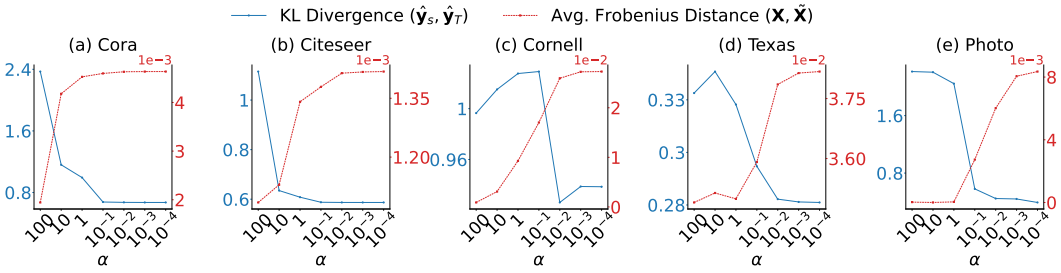

Figure 9: Effect of the hyperparameter α on distillation across CORA, CITESEER, CORNELL, TEXAS, and PHOTO. The KL divergence between student and teacher predictions and the average Frobenius norm between the original and distilled node features are reported for different values of α. The results show that, beyond a certain level of feature change, further changes do not yield additional improvements in KL divergence, indicating diminishing returns from increasing α.

reduction in KL divergence. These results indicate that M2D can encode teacher behavior through relatively small modifications to the original graph and does not require increasingly large data transformations.

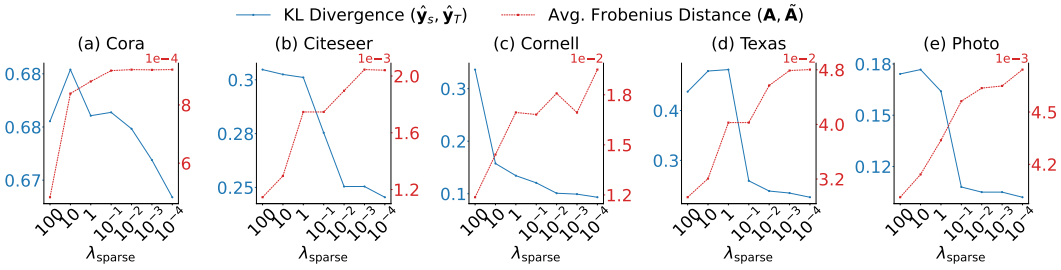

Figure 10: Effect of the hyperparameter λ_{sparse} on distillation across CORA, CITESEER, CORNELL, TEXAS, and PHOTO. The KL divergence between student and teacher predictions and the average Frobenius norm between the original and distilled adjacency matrices are reported for different values of λ_{sparse}. The results show that, beyond a certain level of adjacency change, further changes do not yield additional improvements in KL divergence, indicating diminishing returns from increasing λ_{sparse}.

		GCN				SGC			
		Accuracy ↑	Fidelity ↑	DP ↓	EQOP ↓	Accuracy ↑	Fidelity ↑	DP ↓	EQOP ↓
SIMULATION	Vanilla	96.58 ± 0.14	97.08 ± 0.21	42.42 ± 0.41	4.21 ± 0.37	95.41 ± 0.25	95.83 ± 0.19	40.21 ± 0.29	5.83 ± 0.38
	$\tilde{\mathcal{G}}$(Feat.)	95.47 ± 0.32	97.91 ± 0.19	32.07 ± 0.34	2.59 ± 0.30	95.23 ± 0.51	96.09 ± 0.31	35.47 ± 0.41	$\mathbf{3.91 \pm 0.15}$
	$\tilde{\mathcal{G}}$(Adj)	95.06 ± 0.28	$\mathbf{98.04 \pm 0.31}$	$\mathbf{30.29 \pm 0.27}$	$\mathbf{2.47 \pm 0.15}$	94.98 ± 0.39	$\mathbf{96.72 \pm 0.29}$	$\mathbf{31.27 \pm 0.36}$	4.02 ± 0.27
	$\tilde{\mathcal{G}}$(both)	$\mathbf{96.00 \pm 0.22}$	98.02 ± 0.25	30.70 ± 0.41	2.61 ± 0.37	$\mathbf{95.47 \pm 0.41}$	96.53 ± 0.29	33.05 ± 0.21	3.99 ± 0.31
GERMAN	Vanilla	74.06 ± 0.51	93.21 ± 0.29	9.91 ± 0.53	7.18 ± 0.40	74.01 ± 0.15	92.79 ± 0.23	8.57 ± 0.25	8.01 ± 0.26
	$\tilde{\mathcal{G}}$(Feat)	72.31 ± 0.19	94.57 ± 0.31	4.57 ± 0.26	2.73 ± 0.41	$\mathbf{74.23 \pm 0.32}$	93.58 ± 0.39	5.02 ± 0.43	5.00 ± 0.31
	$\tilde{\mathcal{G}}$(Adj)	$\mathbf{72.45 \pm 0.26}$	94.98 ± 0.14	$\mathbf{3.98 \pm 0.37}$	2.93 ± 0.18	73.97 ± 0.28	93.09 ± 0.24	4.98 ± 0.24	5.71 ± 0.11
	$\tilde{\mathcal{G}}$(Both)	72.38 ± 0.22	$\mathbf{95.02 \pm 0.37}$	4.68 ± 0.23	$\mathbf{2.54 \pm 0.30}$	74.09 ± 0.18	$\mathbf{94.49 \pm 0.35}$	$\mathbf{4.01 \pm 0.34}$	$\mathbf{4.82 \pm 0.18}$

Table 8: Generalization of M2D with a FairGNN teacher on SIMULATION and GERMAN using GCN and SGC as downstream architectures trained only on the augmented graph. Fidelity consistently increases with distillation, indicating improved alignment between student and teacher predictions, while fairness is generally improved, and predictive performance remains comparable.

Tables 8 and 9 evaluate whether the augmented graphs learned by M2D generalize beyond the student architecture used during distillation. We train new downstream models solely on the learned augmented graphs, without access to the teacher during downstream training. With the FairGNN teacher on SIMULATION and GERMAN, both GCN and SGC consistently achieve higher fidelity than their vanilla counterparts, while generally reducing demographic parity and equal opportunity gaps and maintaining comparable predictive performance. Similarly, with the GAT teacher on CITESEER, PHOTO, CORNELL, and TEXAS, feature adjacency and joint augmentation consistently improve the fidelity of downstream GCN and SGC models. These results suggest that M2D encodes teacher behavior into the augmented graph itself rather than tying the transferred knowledge to a particular student architecture, allowing the learned graph to serve as a reusable representation for different downstream models.

		GCN		SGC		MLP	
		Accuracy	Fidelity	Accuracy	Fidelity	Accuracy	Fidelity
CITESEER	Vanilla	71.64 ± 0.16	91.58 ± 0.19	70.01 ± 0.38	91.75 ± 0.30	68.71 ± 0.14	77.25 ± 0.36
	$\tilde{\mathcal{G}}$(Feat.)	$\mathbf{71.98 \pm 0.35}$	93.69 ± 0.33	70.59 ± 0.33	$\mathbf{92.91 \pm 0.31}$	69.29 ± 0.25	79.91 ± 0.24
	$\tilde{\mathcal{G}}$(Adj)	71.70 ± 0.29	93.50 ± 0.19	70.47 ± 0.22	92.41 ± 0.23		
	$\tilde{\mathcal{G}}$(both)	71.87 ± 0.24	$\mathbf{93.91 \pm 0.25}$	$\mathbf{70.91 \pm 0.27}$	92.49 ± 0.29		
PHOTO	Vanilla	90.23 ± 0.23	93.41 ± 0.17	89.23 ± 0.18	92.39 ± 0.23	85.03 ± 0.09	71.41 ± 0.39
	$\tilde{\mathcal{G}}$(Feat)	91.19 ± 0.15	95.47 ± 0.14	90.18 ± 0.31	94.71 ± 0.35	86.91 ± 0.23	73.14 ± 0.24
	$\tilde{\mathcal{G}}$(Adj)	91.03 ± 0.25	95.81 ± 0.41	90.03 ± 0.18	94.59 ± 0.37		
	$\tilde{\mathcal{G}}$(Both)	$\mathbf{91.57 \pm 0.21}$	$\mathbf{96.03 \pm 0.23}$	$\mathbf{91.10 \pm 0.31}$	$\mathbf{94.79 \pm 0.23}$		
CORNELL	Vanilla	53.61 ± 0.41	94.29 ± 0.35	51.52 ± 0.11	89.57 ± 0.23	66.34 ± 0.18	68.24 ± 0.32
	$\tilde{\mathcal{G}}$(Feat)	54.19 ± 0.28	95.87 ± 0.17	52.64 ± 0.27	$\mathbf{93.72 \pm 0.34}$	63.14 ± 0.29	70.17 ± 0.22
	$\tilde{\mathcal{G}}$(Adj)	54.20 ± 0.32	$\mathbf{96.91 \pm 0.20}$	$\mathbf{52.97 \pm 0.28}$	92.21 ± 0.26		
	$\tilde{\mathcal{G}}$(Both)	$\mathbf{54.79 \pm 0.25}$	96.12 ± 0.43	52.39 ± 0.25	92.57 ± 0.26		
TEXAS	Vanilla	62.36 ± 0.20	95.43 ± 0.20	60.29 ± 0.32	91.91 ± 0.32	75.27 ± 0.20	73.18 ± 0.24
	$\tilde{\mathcal{G}}$(Feat)	63.21 ± 0.17	95.86 ± 0.36	61.98 ± 0.21	94.49 ± 0.34	72.29 ± 0.17	75.57 ± 0.18
	$\tilde{\mathcal{G}}$(Adj)	$\mathbf{65.41 \pm 0.15}$	$\mathbf{96.73 \pm 0.27}$	62.30 ± 0.37	$\mathbf{95.03 \pm 0.30}$		
	$\tilde{\mathcal{G}}$(Both)	65.15 ± 0.26	96.08 ± 0.36	$\mathbf{63.59 \pm 0.13}$	94.93 ± 0.19		

Table 9: Generalization of M2D with a GAT teacher on CITESEER, PHOTO, CORNELL and TEXAS using different downstream architectures trained only on the augmented graph. We evaluate M2D with feature and edge augmentation individually and jointly across GCN, SGC, and MLP. Fidelity consistently increases with distillation, indicating improved alignment between the downstream models and teacher predictions.

Table 10 evaluates whether the fairness behavior of a FairGNN teacher can be transferred to an MLP through M2D feature augmentation. Across SIMULATION, GERMAN, and NBA, training the MLP on the augmented features consistently improves fidelity to the teacher while reducing both demographic parity and equal opportunity gaps. On SIMULATION and GERMAN, these fairness gains are achieved with only a modest reduction in accuracy, while on NBA, accuracy also improves. These results show that M2D can encode fairness-related behavior directly into the transformed features, enabling a simple downstream MLP to inherit the teacher's fairer prediction behavior without relying on graph structure.

Figure 11 further examines the KDE distributions of the 20 remaining (see Figure 5) original ($\mathbf{X}$) and learned ($\tilde{\mathbf{X}}$) features across the sensitive groups on the GERMAN dataset. The first and third rows show the distributions of the original features, while the second and fourth rows show their corresponding learned versions after feature masking. Across the features, the learned representa-

		MLP			
		Accuracy (↑)	Fidelity (↑)	DP (↓)	EQOP (↓)
SIMULATION	Vanilla	96.23 ± 0.17	95.91 ± 0.22	42.98 ± 0.33	6.02 ± 0.29
	$\tilde{\mathcal{G}}$(Feat.)	95.02 ± 0.40	97.00 ± 0.25	34.95 ± 0.30	5.21 ± 0.34
GERMAN	Vanilla	76.58 ± 0.22	91.03 ± 0.44	7.39 ± 0.19	8.21 ± 0.29
	$\tilde{\mathcal{G}}$(Feat.)	74.59 ± 0.41	93.47 ± 0.14	5.41 ± 0.25	4.49 ± 0.33
NBA	Vanilla	62.51 ± 0.42	75.23 ± 0.42	6.81 ± 0.61	14.13 ± 0.46
	$\tilde{\mathcal{G}}$(Feat.)	64.25 ± 0.91	77.02 ± 0.81	4.51 ± 0.81	12.12 ± 0.51

Table 10: Generalization of M2D using an MLP student across fairness datasets. We evaluate M2D with feature augmentation. Feature distillation generally improves fidelity and reduces fairness disparities, demonstrating improved alignment with the teacher while maintaining comparable predictive performance.

tions exhibit varying degrees of change, with several features becoming concentrated near zero. This concentration indicates that the learned masks suppress the corresponding feature contributions in the student representation. At the same time, features whose distributions remain similar before and after masking are largely preserved, suggesting that M2D selectively modifies feature contributions rather than uniformly altering the input representation. Together, these results provide further evidence that the learned feature masks adaptively modify the node features to transfer the teacher's behavior to the student.

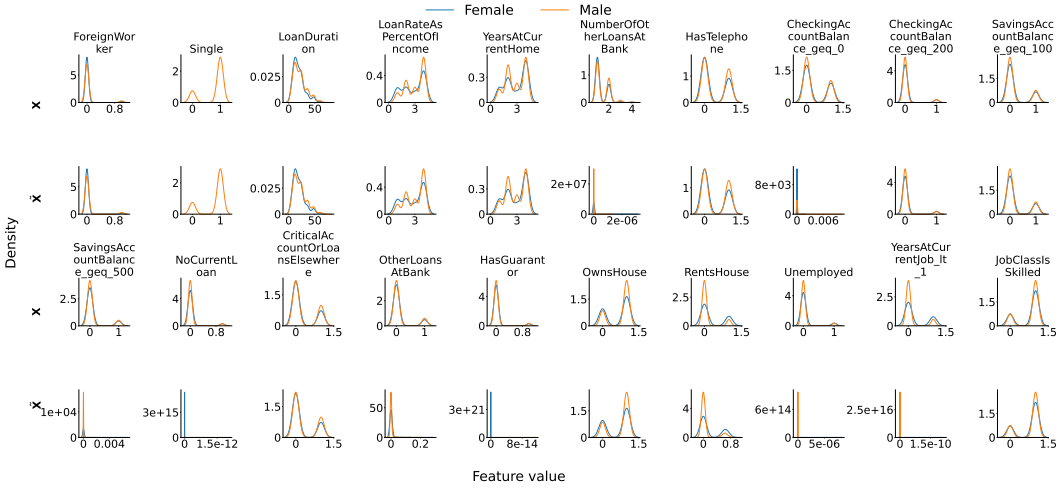

Figure 11: KDE distributions of the 20 remaining (see Figure 5) original ($\mathbf{X}$) and learned ($\tilde{\mathbf{X}}$) features across sensitive groups on the GERMAN dataset. The first and third rows show the original features, while the second and fourth rows show their corresponding learned versions. Features in $\tilde{\mathbf{X}}$ concentrated around zero correspond to features whose contributions are suppressed by the learned masks, thereby reducing their impact on the representation. Teacher: FairGNN, Student: GCN (Feat.).

In Figure 12, we analyze how the learned feature mask aligns with the attention mechanism of the GT teacher by examining the relationship between feature similarity and attention weights. Specifically, we partition node pairs into bins based on similarity $\mathrm{sim}(\mathbf{z}_i, \mathbf{z}_j)$ and compute the average attention within each bin (Similar to Figure 6). In homophilic datasets (CORA, CITESEER, PHOTO), GCN students exhibit a clear positive relationship, indicating that higher attention weights correspond to more similar learned features. In contrast, this relationship weakens in heterophilic datasets (CORNELL, TEXAS), due to lower distillation accuracy (See Table 7). For MLP students, the trend is absent in homophilic datasets due to poor distillation performance but emerges in heterophilic settings where the student better matches the teacher. These results indicate that higher feature similarity is associated with higher attention weights when distillation accuracy is sufficiently high.

Figure 13 shows how structure learning augments a 40-node subgraph from the same dataset. Compared to the original graph (left), the learned graph (right) shows removed or weakened biased

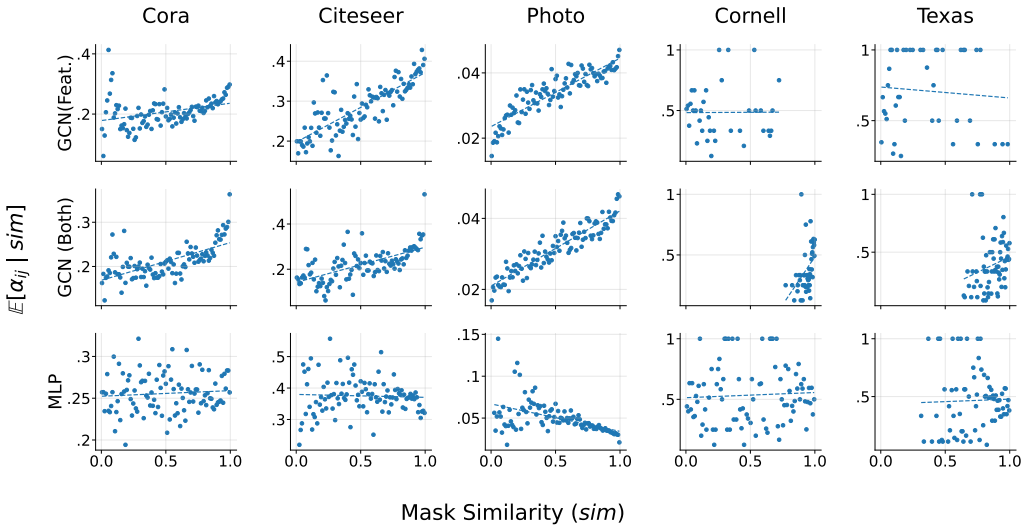

Figure 12: Analysis of learned feature mask with respect to GT attention. We bin mask similarity $\mathrm{sim}(\mathbf{m}_i, \mathbf{m}_j)$ and compute average attention per bin. GCN students exhibit a clear increasing relationship in homophilic datasets (CORA, CITESEER, PHOTO), indicating alignment with teacher attention. This trend weakens in heterophilic datasets due to lower distillation accuracy. For MLP students, the trend appears only when distillation accuracy is high.

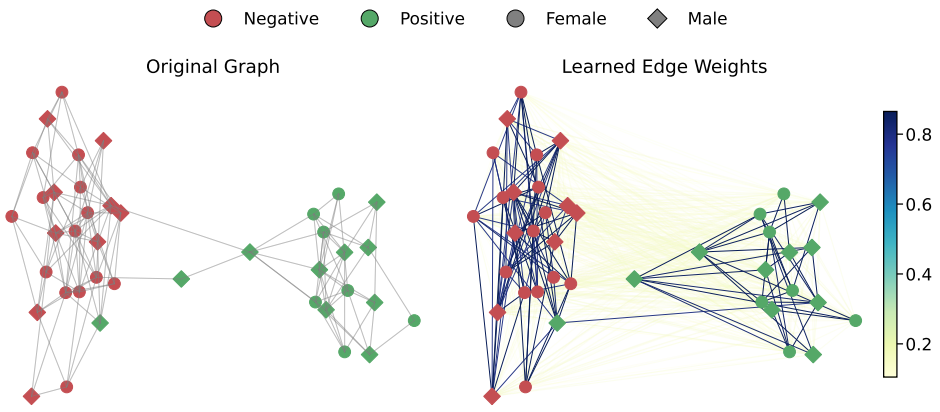

Figure 13: Original and augmented structure for a sample of 40 nodes from the GERMAN dataset, highlighting edges that are modified for fair prediction. Edge weights are encoded by color and width. Node colors denote class labels, and shapes represent sensitive attributes.

connections and strengthened or added unbiased ones. For example, an edge between a male positive node and a male negative node is removed to prevent the male negative node from being biased toward a positive prediction.

COMPLEXITY ANALYSIS

The feature learner takes the node embeddings $\mathbf{H} \in \mathbb{R}^{n \times d_h}$ produced by the GNN and maps them to a feature mask $\mathbf{M} \in \mathbb{R}^{n \times d}$, where d is the dimensionality of the original node features. For a two-layer MLP with hidden dimension d_m, generating the mask requires $\mathcal{O}(nd_h d_m + nd_m d)$ operations. The resulting mask is then applied element-wise to the original feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$. In contrast, the structure learner also takes the node embeddings $\mathbf{H} \in \mathbb{R}^{n \times d_h}$ but generates an edge score for each pair of nodes, resulting in an $n \times n$ adjacency matrix. Consequently, its pairwise edge generation requires $O(n^2 d_h)$ operations for a direct pairwise scoring function. Therefore, the complexity becomes $\mathcal{O}(nd_h d_m + n^2 d_m)$. Thus, feature learning scales linearly with the number of nodes, whereas structure learning scales quadratically with the number of nodes due to its dense pairwise edge generation. The quadratic complexity can be reduced by restricting adjacency learning to a selected subset of node pairs, rather than evaluating all n^2 possible edges, following prior work on scalable graph structure learning Tang et al. (2015); Shin et al. (2023); Saha et al. (2023).

REPRODUCIBILITY

Our code is available in an anonymous GitHub repository: https://anonymous.4open.science/r/m2d-1F9A