A Training Algorithm

The training algorithm for fair graph distillation is shown in Algorithm 1.

Algorithm 1 Fair Graph Distillation

Input: Training graph data $\mathcal{G} = \{\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{S}, \boldsymbol{Y}\}$, hyperparameters α, temperature γ, number of alternative optimization step T_{alt}, distilled label $\boldsymbol{Y}'$.
Initialize $\boldsymbol{X}'$ based on real attributes, synthesizer model ϕ, and GNNs model θ.
for t = 1 to T_{alt} **do**
1. Train GNNs model using distilled graph $\mathcal{G}'_t$ and Equation (15) to obtain GNN_{θ_t} .
2. Given GNNs model GNN_{θ_t}, calculate the gradient distance $D\big(\nabla_\theta \mathcal{L}(\mathcal{G}), \nabla_\theta \mathcal{L}(\mathcal{G}'_t)\big)$ over the real graph $\mathcal{G}$ and distilled graph $\mathcal{G}'_t$.
3. Calculate coherence loss based on GNNs model GNN_{θ_t}, real graph $\mathcal{G}$ and distilled graph $\mathcal{G}'_t$.
4. Train synthesizer model using prediction loss as Equation (16).
end for
Output: The fair distilled graph $\mathcal{G}' = \{\boldsymbol{A}', \boldsymbol{X}', \boldsymbol{Y}'\}$.

B Proof of Theorem 3.4

We consider GNNs model to learn node presentation $\boldsymbol{z}_i$ in the real graph $\mathcal{G}$ and then followed a linear classifier $\boldsymbol{W} = [\boldsymbol{w}_0, \cdots, \boldsymbol{w}_{C-1}]$ and softmax layer, where $\boldsymbol{w}_j$ is the weight vector connected to the j-tj output neuron. We first focus on the relation between the latent representation and the gradient of the linear classification layer. It is easy to obtain the cross-entropy loss J_i (J'_i) for i-th node with label y_i in real graph $\mathcal{G}$ (distilled graph $\mathcal{G}'$) as follows:

$$J_i = -\log \frac{\exp(\boldsymbol{w}_{y_i}^\top \cdot \boldsymbol{z}_i)}{\sum_k \exp(\boldsymbol{w}_k^\top \cdot \boldsymbol{z}_i)}, \tag{17}$$

Then we define gradient over weight vector as $\boldsymbol{g}_{i,j} = \frac{\partial J_i}{\partial \boldsymbol{w}_j}$ and $\boldsymbol{g}'_{i,j} = \frac{\partial J'_i}{\partial \boldsymbol{w}_j}$ in the real and distilled graph. If $j = y_i$, we can obtain

$$\begin{aligned}
\boldsymbol{g}_{i,y_i} &= -\frac{\sum_k \exp(\boldsymbol{w}_k^\top \cdot \boldsymbol{z}_i)}{\exp(\boldsymbol{w}_{y_i}^\top \cdot \boldsymbol{z}_i)} \\
&\quad \cdot \frac{\exp(\boldsymbol{w}_{y_i}^\top \cdot \boldsymbol{z}_i) \sum_k \exp(\boldsymbol{w}_k^\top \cdot \boldsymbol{z}_i) - \exp^2(\boldsymbol{w}_{y_i}^\top \cdot \boldsymbol{z}_i)}{\big(\sum_k \exp(\boldsymbol{w}_k^\top \cdot \boldsymbol{z}_i)\big)^2} \cdot \boldsymbol{z}_i \\
&= -\boldsymbol{z}_i + \frac{\exp(\boldsymbol{w}_{y_i}^\top \cdot \boldsymbol{z}_i)}{\sum_k \exp(\boldsymbol{w}_k^\top \cdot \boldsymbol{z}_i)} \cdot \boldsymbol{z}_i,
\end{aligned} \tag{18}$$

Similarly, for $j \neq y$, we have

$$\boldsymbol{g}_{i,j} = \frac{\exp(\boldsymbol{w}_{y_i}^\top \cdot \boldsymbol{z}_i)}{\sum_k \exp(\boldsymbol{w}_k^\top \cdot \boldsymbol{z}_i)} \cdot \boldsymbol{z}_i, \tag{19}$$

In other words, the gradient of the loss for i-th node with label y_i with respect to the weight vector connected to the j-th output neuron is given by

$$\boldsymbol{g}_{i,j} = \frac{\exp(\boldsymbol{w}_{y_i}^\top \cdot \boldsymbol{z}_i)}{\sum_k \exp(\boldsymbol{w}_k^\top \cdot \boldsymbol{z}_i)} \cdot \boldsymbol{z}_i - \mathbb{1}_{j=y_i} \boldsymbol{z}_i. \tag{20}$$

Based on Assumption 3.1, each model parameter in the last softmax layer satisfies the same distribution. In other words, the expectation of all predictions are the same, i.e.,

$$\mathbb{E}_{\mathcal{P}_\theta}\Big[\frac{\exp(\boldsymbol{w}_0^\top \cdot \boldsymbol{z}_i)}{\sum_k \exp(\boldsymbol{w}_k^\top \cdot \boldsymbol{z}_i)}\Big] = \cdots = \mathbb{E}_{\mathcal{P}_\theta}\Big[\frac{\exp(\boldsymbol{w}_{C-1}^\top \cdot \boldsymbol{z}_i)}{\sum_k \exp(\boldsymbol{w}_k^\top \cdot \boldsymbol{z}_i)}\Big]. \tag{21}$$

Note that the gradient calculation is based on backpropagation, the gradient for the last linear classification layer is quite critical for the gradient of other layers. Hence we consider the gradient of the last linear classification layer in the real graph, shown by

$$\begin{aligned}\mathbb{E}_{\theta\sim\mathcal{P}_\theta}\left[\nabla_{\boldsymbol{w}_j}\mathcal{L}(\mathcal{G})\right] &= \mathbb{E}_{\theta\sim\mathcal{P}_\theta}\left[\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{g}_{i,j}\right] \\ &= \frac{1}{NC}\sum_{i=1}^{N}\boldsymbol{z}_i - \frac{1}{N}\sum_{\{i:y_i=j\}}\boldsymbol{z}_i,\end{aligned} \tag{22}$$

Similarly, we have the gradient of the last linear classification layer in the distilled graph as follows:

$$\begin{aligned}\mathbb{E}_{\theta\sim\mathcal{P}_\theta}\left[\nabla_{\boldsymbol{w}_j}\mathcal{L}(\mathcal{G}')\right] &= \mathbb{E}_{\theta\sim\mathcal{P}_\theta}\left[\frac{1}{N'}\sum_{i=1}^{N}\boldsymbol{g}'_{i,j}\right] \\ &= \frac{1}{N'C}\sum_{i=1}^{N'}\boldsymbol{z}'_i - \frac{1}{N'}\sum_{\{i:y'_i=j\}}\boldsymbol{z}'_i,\end{aligned} \tag{23}$$

Under assumption 3.2, it is easy to know that the optimal solution to minimizing the objective $\min_{\mathcal{G}'}\mathbb{E}_{\theta\sim\mathcal{P}_{\boldsymbol{W}}}\left[||\nabla_{\boldsymbol{W}}\mathcal{L}(\mathcal{G})-\nabla_{\boldsymbol{W}}\mathcal{L}(\mathcal{G}')||^2\right]$ satisfy $\nabla_{\boldsymbol{W}}\mathcal{L}(\mathcal{G}) = \nabla_{\boldsymbol{W}}\mathcal{L}(\mathcal{G}')$. Since the distilled label is sampling to keep class label probability, we have $\frac{|\{i:y'_i=j\}|}{N'} = \frac{|\{i:y_i=j\}|}{N}$ for any class index i. Therefore, based on Equations (22) and (23), we have the optimal distilled graph satisfy

$$\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{z}_i = \frac{1}{N'}\sum_{i=1}^{N'}\boldsymbol{z}'_i. \tag{24}$$

C Proof of Ridge Regression

Define objective function $J = \gamma\|\boldsymbol{z}' - \boldsymbol{Z}_s^\top\boldsymbol{q}\|_2^2 + \|\boldsymbol{q}\|_2^2$, it is easy to obtain

$$\frac{\partial J}{\partial \boldsymbol{q}} = -2\gamma\boldsymbol{Z}_s\left(\boldsymbol{z}' - \boldsymbol{Z}_s^\top\boldsymbol{q}\right) + 2\boldsymbol{q} = 0 \tag{25}$$

Therefore, the optimal $\boldsymbol{q}^* = \gamma(\boldsymbol{I} + \gamma\boldsymbol{Z}_s\boldsymbol{Z}_s^\top)^{-1}\boldsymbol{Z}_s\boldsymbol{z}'$. Therefore, the projection of representation $\boldsymbol{z}'$ in the complement space of sensitive group $\boldsymbol{Z}_s$ is given by

$$\boldsymbol{z}' - \boldsymbol{Z}_s^\top\boldsymbol{q}^* = \boldsymbol{z}' - \gamma\boldsymbol{Z}_s^\top(\boldsymbol{I} + \gamma\boldsymbol{Z}_s\boldsymbol{Z}_s^\top)^{-1}\boldsymbol{Z}_s\boldsymbol{z}' \tag{26}$$

D More Results on Consistent Span Space

We conduct experiments to measure the distance between $span(Z)$ and $span(Z')$ using principle angles between subspaces and emperically shows that $span(Z) \approx span(Z')$ in the real dataset.

The concept of principal angle is used in linear algebra to measure the similarity between two subspaces of a vector space. It helps quantify how close or far apart these subspaces are. Given subspace, $\mathbf{L}, \mathbf{M} \subseteq \mathbb{R}^n$, with $\dim \mathbf{L} = l \geq \dim \mathbf{M} = m$, there are m principal angles between L and M denoted as $0 \leq \theta_1 \leq \theta_2 \leq \cdots \leq \theta_m \leq \frac{\pi}{2}$ between L and M are recursively defined, where

$$\cos(\theta_i) := \min\left\{\frac{<\mathbf{x},\mathbf{y}>}{\|\mathbf{x}\|\|\mathbf{y}\|} \mid \mathbf{x}\in\mathbf{L}, \mathbf{y}\in\mathbf{M}, \mathbf{x}\perp\mathbf{x}_k, \mathbf{y}\perp\mathbf{y}_k, k = 1,\cdots,i-1\right\}. \tag{27}$$

Notably, when the two subspaces are aligned, the principal angels are close to 0. We report the average principal angles of $span(Z)$ and $span(Z')$ on all datasets as following:

- Pokec-z: 1.08×10^{-6}
- Pokec-n: 1.03×10^{-6}
- German: 4.84×10^{-7}

- Credit: 2.57×10^{-7}
- Recidivism: 3.87×10^{-7}

In the experiments, the principal angles of $span(Z)$ and $span(Z')$ on all dataset are nearly 0. This indicates that the distance between space $span(Z')$ and space $span(Z)$ are quite small in practice.

Additionally, we would like to mention that [1] provides the rigorous proof of $z' \in span(Z)$ for distribution matching under several assumptions (although we can not prove it under gradient matching setting). According to formulas 21 from [1], it is assumed that (1) the **linear extractor** $\psi_{\boldsymbol{\theta}} : \mathbb{R}^d \to \mathbb{R}^k$ such that $k < d, \boldsymbol{\theta} = [\theta_{i,j}] \in \mathbb{R}^{k \times d}, \theta_{i,j} \overset{iid}{\sim} \mathcal{N}(0,1)$ and for an input $\mathbf{z}$, $\psi_{\boldsymbol{\theta}}(\mathbf{x}) = \boldsymbol{\theta}\mathbf{z}$. When using distribution match method for data condensation, we have:

$$\frac{\partial L}{\partial \mathbf{z}_i'} = \frac{\partial E_\theta ||d||^2}{\partial \mathbf{z}_i'} = -\frac{2}{|N'|} \left(\frac{1}{N} \sum_{j=1}^{N} \mathbf{z}_j - \frac{1}{N'} \sum_{j=1}^{N'} \mathbf{z}_j' \right)^T \cdot E[\boldsymbol{\theta}^t \boldsymbol{\theta}]$$

where $d := \theta\left(\frac{1}{N}\sum_{j=1}^{N} \mathbf{z}_j - \frac{1}{N'}\sum_{j=1}^{N'} \mathbf{z}_j'\right)$, $\mathbb{E}\left[\boldsymbol{\theta}^\top \boldsymbol{\theta}\right] = k\mathbf{I}_d$ by definition of $\boldsymbol{\theta}$, and $\mathbf{I}_d$ is the identity matrix of $\mathbb{R}^d$. the projection components of $\mathrm{span}(Z)^\perp$ remain zero throughout the optimization process of DM. And we use $\mathbf{z}_i$ to initialize $\mathbf{z}_i'$, thus $\mathbf{z}' \in span(Z)$. However, in the implementation we use gradient matching instead of distribution matching.

Table 3: Statistical Information on Datasets

Dataset	# Nodes	# Attributes	# Edges	Avg. degree	Sens	Label
Pokec-n	6,185	59	21,844	7.06	Region	Working field
Pokec-z	7,659	59	29,476	7.70	Region	Working field
German	1,000	27	21,242	44.50	Gender	Credit status
Credit	30,000	13	1,436,858	95.80	Age	Future default
Recidivism	18,876	18	321,308	34.00	Race	Bail decision

E Preliminary Motivation

We have added experiments comparing the fairness performance of various fair GNNs trained on synthetic and real graph data. Specifically, we report the results (using demographic parity (DP), equal opportunity (EO), and individual unfairness (IND) Song et al. [2022] as metrics) with EDITS Dong et al. [2022], FairGNN Dai and Wang [2021], InFoRM Kang et al. [2020], and REDRESS Dong et al. [2021] on five datasets in our paper. EDITS is a pre-processing debiasing method, FairGNN is an in-processing debiasing method, and InFoRM and REDRESS focus on individual fairness. We encountered out-of-memory (OOM) issues when implementing GUIDE and REDRESS on an NVIDIA GeForce RTX A5000 (24GB GPU memory), so we used InFoRM as the baseline. Due to the extensive training time required for REDRESS, we only report results on the German dataset for REDRESS. We use demographic parity (DP), equal opportunity (EO), and individual unfairness (IND) as metrics.Table 4 demonstrates the result. From Table 4, we can see that in terms of the group fairness metrics (DP, EO), the fairness problem becomes uniformly worse on the Credit, German, and Pokecn datasets for all debiasing methods. For the Recidivism dataset, the distilled graph shows fewer fairness issues (lower DP or EO), especially for the EDITS method. This may result from the drop in utility of the model trained on the distilled graph (AUC is too low). As shown in Figure 4 of our paper, FGD can achieve a better performance-fairness trade-off compared to the baselines.

F Dataset Statistics

Pokec. The Pokec dataset consists of millions of anonymized user profiles from Slovakia's most popular social network in 2012, with information such as gender, age, hobbies, interests, education, and working field. The dataset was sampled into Pokec-z and Pokec-n based on user province, with region as the sensitive attribute. The task is to predict user working field.

Table 4: Utility and group fairness comparison between real graph and distilled graph with various debias method. **Bold** value indicates worse fairness performance.

		Recidivism		Credit		German		Pokecn		Pokecz	
		Real	Distillated	Real	Distillated	Real	Distillated	Real	Distillated	Real	Distillated
EDITS	AUC↑	0.971	0.658	0.740	0.704	0.668	0.506	OOM	OOM	OOM	OOM
	DP↓	0.067	0.005	0.027	**0.063**	0.009	**0.024**	OOM	OOM	OOM	OOM
	EO↓	0.038	0.011	0.018	**0.028**	0.008	**0.030**	OOM	OOM	OOM	OOM
FairGNN	AUC↑	0.977	0.788	0.759	0.720	0.742	0.645	0.782	0.676	0.784	0.723
	DP↓	0.065	0.046	0.062	**0.123**	0.010	**0.013**	0.005	**0.044**	0.042	**0.037**
	EO↓	0.037	0.046	0.037	**0.091**	0.001	**0.011**	0.006	**0.062**	0.051	**0.038**
InFoRM	AUC↑	0.906	0.708	0.741	0.717	0.642	0.538	0.743	0.644	0.751	0.708
	DP↓	0.011	0.118	0.004	**0.174**	0.085	0.018	0.009	0.009	0.020	**0.048**
	EO↓	0.024	0.092	0.001	**0.135**	0.153	0.017	0.013	0.015	0.018	**0.038**
	IND↓	8098	**3596022**	2699	**338149**	4360	**24888**	6466	**272013**	6828	**199853**
REDRESS	AUC↑	OOM	OOM	OOM	OOM	0.719	0.483	OOM	OOM	OOM	OOM
	DP↓	OOM	OOM	OOM	OOM	0.005	**0.043**	OOM	OOM	OOM	OOM
	EO↓	OOM	OOM	OOM	OOM	0.010	**0.073**	OOM	OOM	OOM	OOM
	IND↓	OOM	OOM	OOM	OOM	9728	**186366**	OOM	OOM	OOM	OOM

Table 5: Parameter study of α. All the value is in scale of $\times 10^2$.

α	AUC	Δ_{DP}	Δ_{EO}	Bias
0.04	74.75	0.84	0.88	0.19
0.5	69.42	0.66	0.43	0.15
0.6	69.37	0.58	0.16	0.14
1.0	65.35	0.00	0.00	0.11

German. The German Graph credit dataset has 1,000 client records with attributes like Gender and LoanAmount, used to classify individuals as good or bad credit risks. The similarity between node attributes is calculated using Minkowski distance and nodes are connected if the similarity is 80% of the maximum similarity.

Credit. Credit dataset, consisting of 30,000 individuals with features such as education, credit history, age, and derived spending and payment patterns. The similarity between two node attributes is calculated using Minkowski distance as the similarity measure and the credit defaulter graph network is constructed by connecting nodes with a similarity of 70% of the maximum similarity between all nodes.

Recidivism. The US state court bail outcome dataset (1990-2009) contains 18,876 defendant records with past criminal records, demographic attributes, etc. The similarity between node attributes is calculated using Minkowski distance and nodes are connected if the similarity is 60% of the maximum similarity.

G More Experimental Details

G.1 Parameter Study

Here we aim to study the sensitivity of FGD w.r.t. hyper-parameters. Specifically, we show the parameter study of α on Recidivism dataset. Here α controls the intensity to regularize the coherence bias of the distilled small graph. The results in Table 5 indicate that α can control the debiasing and utility performance of the distilled small graph.

G.2 Implementation Details

Synthesizer training. We adopt Adam optimizer for synthesizer training with 0.0002 learning rate. MLP_ϕ consists of 3 linear layer with 128 hidden dimension. The outer loop number is 16 while the inner loop is 4 for each epoch. For each experiment, we train with a maximum of 1200 epochs and 3 independent runs. The temperature parameter γ is set to 10. $\boldsymbol{X}'$ and ϕ are optimized alternatively.

GNN training. We adopt Adam optimizer for GNN training with 0.005 learning rate. All GNN models are 2 layers with 256 hidden dimensions. For Pokec-z, Pokec-n, German, Credit, and Recidivism the training epochs are 1500, 1500, 4000, 1000, and 1000 respectively.

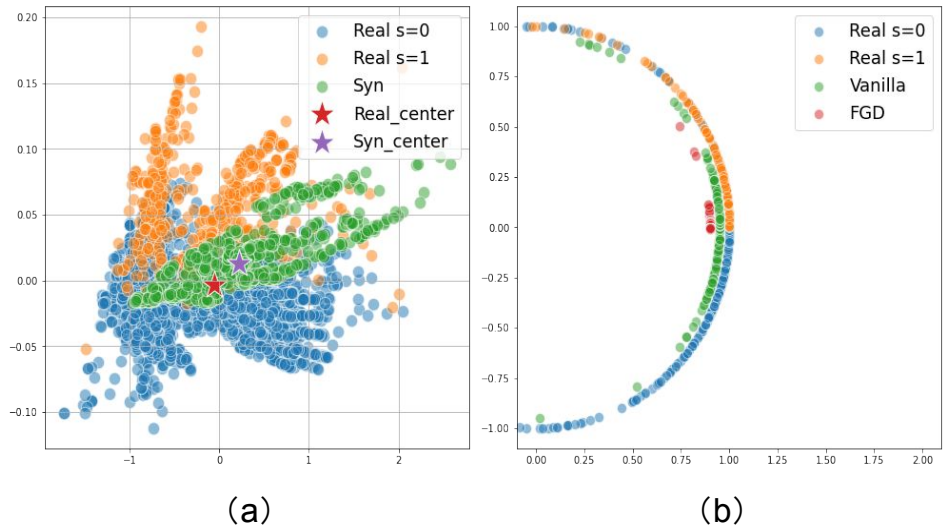

Figure 6: (a) shows the visualization of node representations from real graph and distilled graph, as well as their barycenter, on Credit dataset, after PCA. (b) shows the visualization of geometric intuition of node from real graph and distilled graph on Credit dataset.

G.3 More Visualization

We also visualize the node representation using PCA. We could observe that the barycenter of node from real graph and distilled graph is very close. And The distribution of node representation after being normalized to the circumference is consistent with the geometric intuition shown in Figure 6.

H Limitations and Future Work

H.1 Non-binary Sensitive Attribute

For categorical sensitive attributes, if only one sensitive membership group's embeddings are far away from others, then the mean embeddings will still be close to the majority embeddings, especially for many categories, resulting in low variance (coherence). We argue that only this group with distant embedding (a small portion of samples) can have their sensitive attribute detected using embedding distributions. From a metric perspective, if we adopt the maximized Δ_{DP} over any sensitive attribute group pair, the bias should be large due to considering the worst case. The proposed coherence may not work well in this scenario, and an advanced coherence can be developed for this case, e.g., the maximized variance over any sensitive group pair. We leave the advanced coherence development for categorical, multiple, or even continuous sensitive attributes in future work.

H.2 Individual Fairness

From Table 4, we find that all datasets suffer from a surprisingly more severe individual fairness problem (much higher IND score) when the model is trained on the distilled graph, even if we use InFoRM or REDRESS. This could be an interesting direction for future work, and we will add discussion with references in the related work section.

H.3 Other Tasks

Our paper mainly focuses node classification tasks and it is possible to extend our method to other tasks or other group fairness problems. For instance, FGD may alleviate group fairness issues in link prediction tasks by reducing the coherence bias among different link groups. Exploring other tasks (e.g., recommendation, graph classification) or other fairness metrics (e.g., individual fairness, rank fairness) could be interesting for future work.