Appendix A: Graph Neural Networks

The update process for the l-th layer in a GNN is defined by two main steps: aggregation and combination. Here's a detailed explanation of each step:

$$\mathbf{a}_i^{(l)} = \text{AGGREGATE}^{(l-1)}\left(\left\{\mathbf{h}_j^{(l-1)} : j \in \mathcal{N}(v_i)\right\}\right). \tag{1}$$

In this step, the GNN aggregates the representation vectors of the neighbors of node v_i from the previous layer $(l-1)$. The function $\text{AGGREGATE}^{(l-1)}$ can be any operation that combines the vectors, such as summing, averaging, or using more complex functions like max pooling or attention mechanisms. The result, $\mathbf{a}_i^{(l)}$, is a vector that captures the neighborhood information of node v_i at the l-th layer.

$$\mathbf{h}_i^{(l)} = \text{COMBINE}^{(l)}\left(\mathbf{h}_i^{(l-1)}, \mathbf{a}^{(l)}\right). \tag{2}$$

In this step, the GNN combines the representation vector of node v_i from the previous layer $(l-1)$ with the aggregated neighborhood vector $\mathbf{a}_i^{(l)}$. The function $\text{COMBINE}^{(l)}$ can be a simple concatenation followed by a linear transformation, or a more complex operation like a gated mechanism or a non-linear transformation. The result, $\mathbf{h}_i^{(l)}$, is the updated representation vector of node v_i at the l-th layer.

In this paper, we use GraphSAGE as the backbone for all fair GNNs. The complete graph sampling utilized by GCN has been optimized by GraphSAGE through partial node-centric neighbor sampling. This optimization facilitates the distributed training of extensive amounts of graph data. GraphSAGE advances prior work in the inductive setting by equally emphasizing each neighbour node. The convolutional propagation rule employed in the GCN architecture is remarkably identical to the mean aggregator chosen for GraphSAGE. The formula of GraphSAGE is defined as

$$\mathbf{h}_{\mathcal{N}(v)}^{(l)} \leftarrow \mathbf{h}_u^{(l-1)}, \forall u \in \mathcal{N}(v), \tag{3}$$

$$\mathbf{h}_v^{(l)} \leftarrow \sigma\left(\mathbf{W}^{(l)} \cdot \text{MEAN}\left(\left\{\mathbf{h}_v^{(l-1)}\right\} \cup \mathbf{h}_{\mathcal{N}(v)}^{(l)}\right)\right), \tag{4}$$

where $\mathbf{h}_v^l$ denotes the features of the node v on the l-th layer, $\mathcal{N}(v)$ represents the neighborhood of the node v, and $\text{MEAN}(\cdot)$ is to compute the mean of the features with the neighborhood in this paper.

Appendix B: Fairness Evaluation Metrics

A fundamental principle of fair machine learning is to ensure equal treatment for both majority and minority groups, a concept known as group fairness. In this paper, we focus on group fairness and use two quantitative metrics to evaluate it: Δ_{SP} and Δ_{EO}. Δ_{SP} measures the statistical parity differences between groups, while Δ_{EO} evaluates the differences in equalized odds. Lower values for both metrics indicate greater fairness in the model's predictions across different groups. We use Δ_{SP} and Δ_{EO} to systematically evaluate group fairness. Importantly, both metrics allow for a rigorous assessment of whether the model achieves fair treatment for minorities.

Definition 1 (Statistical Parity)**.** *Statistical parity stipulates that the proportion of individuals receiving positive classifications is approximately equalized across demographic groups:*

$$P(\hat{y} \mid s=0) = P(\hat{y} \mid s=1), \tag{5}$$

where $\hat{y}$ represents the prediction, and s denotes the sensitive attribute, such as race or gender.

Definition 2 (Equal Opportunity)**.** *Equal opportunity requires that the true positive rate be roughly the same across different demographic groups:*

$$P(\hat{y}=1 \mid y=1, s=0) = P(\hat{y}=1 \mid y=1, s=1) \tag{6}$$

where y represents the true label.

Based on Definition 1 and Definition 2, the fairness metrics Δ_{SP} and Δ_{EO} can be calculated as:

$$\Delta_{SP} = |P(\hat{y}=1 \mid s=0) - P(\hat{y}=1 \mid s=1)|, \tag{7}$$

$$\Delta_{EO} = |P(\hat{y}=1 \mid y=1, s=0) - P(\hat{y}=1 \mid y=1, s=1)|. \tag{8}$$

The choice of Δ_{SP} and Δ_{EO} provides two complementary views on fairness, which balances outcome rates and prediction errors across groups. Together, they provide a comprehensive methodology for auditing and ensuring group fairness.

Appendix C: Hyper-parameter Analysis

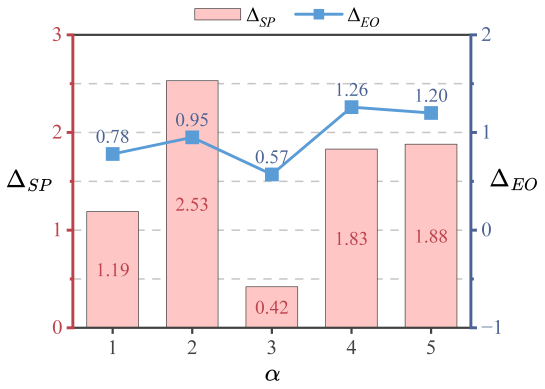


Figure 1: Fairness performance under different values of α.

In the counterfactual data generation module, a critical parameter is the KNN K-value α used for adjacency matrix reconstruction. The experimental results, as shown in the Fig. 1, provide insights into the optimal values of α for different datasets on the German dataset. For datasets with a higher average node degree (Credit and German), a smaller K-value (α = 3) is effective. For datasets with a lower average node degree (e.g., Bail), a larger K-value (α = 6) is necessary. This approach ensures that the reconstructed adjacency matrix is sufficiently different from the original, thereby effectively intervening on sensitive factors and improving the fairness and utility of the learned representations. This finding underscores the importance of considering both attribute and structural changes when generating counterfactual data for fair graph representation learning.

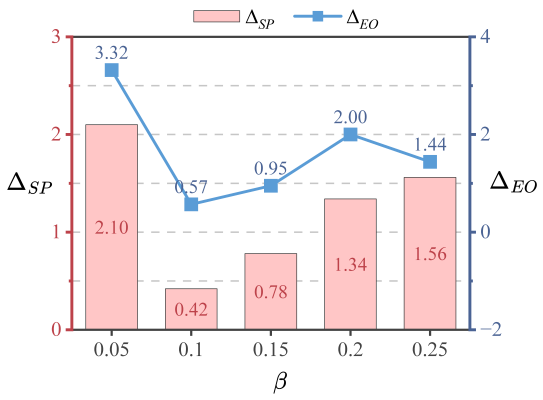
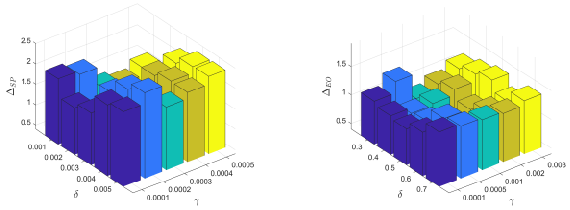


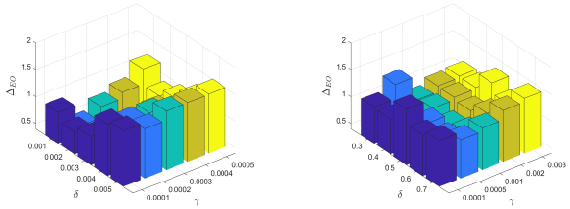
Figure 2: Fairness performance under different values of β.

The hyperparameter β represents the weight of the fairness constraint for the causal representation. The experimental results for β on the German are shown in Fig. 2. For all three datasets, the model achieves the best fairness metrics when β = 0.2. When β is set to a larger value, it can lead to suboptimal disentanglement between causal and sensitive factors. This results in a decrease in fairness performance.



(a) Δ_{SP} (b) Δ_{EO}

Figure 3: Fairness performance under different values of γ and δ on German.



(a) Credit (b) Bail

Figure 4: Fairness performance on Δ_{EO} under different values of γ and δ on Credit and Bail.

In the factor disentanglement module, the hyperparameters γ and δ play crucial roles in achieving effective disentanglement of factors. The experimental results for these hyperparameters on the German dataset are shown in Fig 3 and Fig 4. For the Credit and German datasets, which are relatively dense, lower values of γ (0.002) and δ (0.0003) are sufficient to achieve effective disentanglement. For the Bail dataset, which is relatively sparse, larger values of γ (0.5) and δ (0.001) are required to achieve effective disentanglement. This may signify that causal and sensitive factors in the representation of sparse data are more difficult to disentangle.

Dataset	**Metric**	0.3	0.5	0.7
Credit	ACC (↑)	76.77±1.59	77.74±0.27	77.18±1.74
	F1 (↑)	86.18±1.73	87.32±0.25	86.52±1.60
	$\Delta_{SP}(\downarrow)$	2.56±2.84	0.44±0.53	1.45±1.16
	$\Delta_{EO}(\downarrow)$	1.84±1.80	0.44±0.53	0.99±0.69
German	ACC (↑)	69.04±1.90	69.84±0.20	69.76±0.60
	F1 (↑)	80.98±2.22	82.17±0.22	82.10±0.31
	$\Delta_{SP}(\downarrow)$	1.59±1.01	0.42±0.75	1.13±0.63
	$\Delta_{EO}(\downarrow)$	1.24±0.74	0.57±1.08	1.05±0.62
Bail	ACC (↑)	88.32±1.46	81.62±0.53	88.62±1.02
	F1 (↑)	83.40±1.99	85.60±1.92	84.19±1.35
	$\Delta_{SP}(\downarrow)$	1.33±1.02	0.53±0.43	1.24±0.73
	$\Delta_{EO}(\downarrow)$	1.24±0.73	0.83±0.39	1.13±0.57

Table 1: Performance under different values of κ.

We can observe from the experimental results as shown in Table 1 that when 30% of the dimensions are allocated to causal factors, the fairness metrics become unstable. This indicates that the causal factors and sensitive factors are not stably disentangled. When 70% of the dimensions are allocated to causal factors, the fairness metrics deteriorate, suggesting that the sensitive factors are not sufficiently captured. Meanwhile, we can see that when 50% of the dimensions are assigned to causal factors, both the fairness metrics and the F1 score achieve the best performance. This demonstrates that the causal factors and sensitive factors are effectively disentangled and fully utilized.