RawlsGCN: Towards Rawlsian Difference Principle on Graph Convolutional Network

Jian Kang¹
Yan Zhu²
Yinglong Xia²
Jiebo Luo²,³
Hanghang Tong¹

¹ University of Illinois at Urbana-Champaign
² Meta AI
³ University of Rochester
Ubiquity of Graphs

Social Network Analysis
Drug Discovery
Recommendation
Traffic Prediction
Fraud Detection
Question Answering

This Presentation: Graph = Network
Graph Convolutional Network (GCN)

- **Key idea:** Learn node representations by aggregating information from the neighbors – a.k.a. graph convolution

- **GCN:** A stack of graph convolution layers
  \[
  H^{(l)} = \sigma(\hat{A}H^{(l-1)}W^{(l)})
  \]

  \(\hat{A} = \tilde{D}^{-\frac{1}{2}}(A + I)\tilde{D}^{-\frac{1}{2}}\)

  \(\tilde{D} = \text{degree matrix of } A + I\)

[1] Kipf, T. N., & Welling, M.. Semi-Supervised Classification with Graph Convolutional Networks. ICLR 2017.
Degree-related Unfairness

• **Observation:** Low-degree node often has
  – High loss
  – Low predictive accuracy

• **Example:** Semi-supervised node classification

![Graphs showing relationship between node degree, average accuracy, and average loss.](image)
Degree-related Unfairness

- **Example:** Online advertising
  - Celebrities often enjoy high-quality recommendations
  - Grassroot users often suffer from bad recommendations
Degree Distribution

- Node degree distribution is often long-tailed

- GCN might
  - Benefit a relatively small fraction of high-degree nodes
  - Overlook a relatively large fraction of low-degree nodes

[1] Faloutsos, M., Faloutsos, P., & Faloutsos, C. On Power-Law Relationships of the Internet Topology. CCR 1999.
Prior Works

• DEMO-Net
  – **Degree-specific weight**: Learn degree-specific weights, randomly initialized

• SL-DSGCN
  – **Degree-specific weight**: Learn degree-specific weights, generated by RNN
  – **Self-supervised learning**: Generate pseudo labels for additional training signals

• Tail-GNN
  – **Neighborhood translation mechanism**: Infer missing neighborhood information of low-degree nodes

• **Limitation 1**: Additional number of weight parameters
  – DEMO-Net, SL-DSGCN

• **Limitation 2**: Change(s) to the GCN architecture
  – SL-DSGCN, Tail-GNN

• **Question**: How to mitigate degree-related unfairness without
  – Hurting the scalability of GCN
  – Changing the GCN architecture?

---

[1] Wu, J., He, J., & Xu, J.. DEMO-Net: Degree-Specific Graph Neural Networks for Node and Graph Classification. KDD 2019.
[2] Tang, X., Yao, H., Sun, Y., Wang, Y., Tang, J., Aggarwal, C., ... & Wang, S.. Investigating and Mitigating Degree-Related Biases in Graph Convolutional Networks. CIKM 2020.
[3] Liu, Z., Nguyen, T. K., & Fang, Y.. Tail-GNN: Tail-Node Graph Neural Networks. KDD 2021.
Fairness = Just Allocation of Utility

- **Intuition:** Utility = resource to allocate
- **Expected result:** Similar utility (accuracy) for all nodes regardless of their degrees
- **Example**

![Graphs showing fairness in utility allocation](image-url)
Example: Fair Allocation of Utility

- **Example:** Fair online advertising

- **Question:** How to define such fairness?

Debiasing
Problem Definition

• Given
  – An undirected graph $\mathcal{G} = (A, X)$
  – An $L$-layer GCN with weights $\theta$
  – A task-specific loss $J$

• Find: A well-trained GCN that
  – Minimizes the task-specific loss
  – Achieves a fair allocation of utility for the groups of nodes with the same degree

• Key question: When is the allocation of utility fair?
Rawlsian Difference Principle

- **Origin:** Distributive justice
- **Goal:** Find a fair allocation of social welfare

“Inequalities are permissible when they maximize [...] the long-term expectations of the least fortunate group.”

-- John Rawls, 1971

- **Intuition:** Treat utility of GCN as welfare to allocate
  - Least fortunate group → group with the smallest utility
  - **Example:** Classification accuracy for node classification

[1] Rawls, J.. A Theory of Justice. Press, Cambridge 1971.

- **Justice as fairness**
  - Justice is a virtue of institutions
  - Free persons enjoy and acknowledge the rules
- **Well-ordered society**
  - Designed to advance the good of its members
  - Regulated by a public conception of justice
Key Challenge: Fair Allocation of Utility

- **Key idea:** Consider the stability of the Rawlsian difference principle
- **How to achieve the stability?**
  - Keep improving the utility of the least fortunate group
- **When do we achieve the stability?**
  - No least fortunate group
  - All groups have the balanced utility
- **Challenge:** Non-differentiable utility
  - **Workaround:** Use loss function as the proxy of utility
  - **Rationale:** Minimize loss in order to maximize utility
- **Goal:** Fair allocation of utility → balanced loss
Roadmap

- Motivation
- Theory: Source of Unfairness
- Algorithms: RawlsGCN
- Experiments
- Conclusion
Theory: Source of Unfairness

- **Intuition:** Understand why the loss varies after training

- **What happens during training?**
  - Extract node representations
  - Predict the outcomes using the node representations
  - Calculate the task-specific loss $J$
  - Update model weights $\theta$ by the gradient $\frac{\partial J}{\partial \theta}$ ← key component for training

- **Question:** Is the unfairness caused by the gradient?
The Gradient of Model Weights

• Given
  – An undirected graph \( \mathcal{G} = (\mathbf{A}, \mathbf{X}) \) with \( \tilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{A} + \mathbf{I})\mathbf{D}^{-\frac{1}{2}} \)
  – An arbitrary \( l \)-th graph convolution layer
    • Weight matrix \( \mathbf{W}^{(l)} \)
    • Hidden representations before activation \( \mathbf{E}^{(l)} = \tilde{\mathbf{A}}\mathbf{H}^{(l-1)}\mathbf{W}^{(l)} \)
  – A task-specific loss \( J \)

• The gradient of loss \( J \) w.r.t. weight \( \mathbf{W}^{(l)} \)

\[
\frac{\partial J}{\partial \mathbf{W}^{(l)}} = \left(\mathbf{H}^{(l-1)}\right)^T \tilde{\mathbf{A}}^T \frac{\partial J}{\partial \mathbf{E}^{(l)}}
\]

\[
\frac{\partial J}{\partial \mathbf{W}^{(l)}} = \left(\mathbf{H}^{(l-1)}\right)^T \tilde{\mathbf{A}}^T \frac{\partial J}{\partial \mathbf{E}^{(l)}}
\]
Source of Unfairness: Results

- \( \frac{\partial J}{\partial W^{(l)}} \) is a linear summation of node influence weighted by its degree in \( \hat{A} \)

\[
\frac{\partial J}{\partial W^{(l)}} = \sum_{i=1}^{n} d_{\hat{A}}(i)I_{i}^{(\text{col})} = \sum_{j=1}^{n} d_{\hat{A}}(j)I_{j}^{(\text{row})}
\]

- \( I_{i}^{(\text{col})} = \left( E_{j \sim N(i)} \left[ H^{(l-1)}[j,:) \right] \right)^T \frac{\partial J}{\partial E^{(l)}[i,:]} \)
- \( I_{j}^{(\text{row})} = \left( H^{(l-1)}[j,:) \right)^T E_{i \sim \hat{N}(j)} \left[ \frac{\partial J}{\partial E^{(l)}[i,:]} \right] \)
- \( j \sim \hat{N}(i) \): Sampling node \( j \) from neighborhood of node \( i \) in \( \hat{A} \)
  - Sampling probability is proportional to \( \hat{A}[i,j] \)

\[
d_{\hat{A}}(1) = 2
\]
\[
d_{\hat{A}}(2) = 1
\]
\[
d_{\hat{A}}(3) = 1
\]

\[
\frac{\partial J}{\partial W^{(l)}} = 2I_{1}^{(\text{col})} + I_{2}^{(\text{col})} + I_{3}^{(\text{col})}
\]

Higher importance due to higher degree
Source of Unfairness: Column-wise Influence

\[ \frac{\partial J}{\partial W^{(l)}} \] is a linear summation of node influence weighted by its degree in \( \hat{A} \)

\[ \frac{\partial J}{\partial W^{(l)}} = \sum_{i=1}^{n} d_{\hat{A}}(i) \mathbb{I}^{(\text{col})}_i = \sum_{j=1}^{n} d_{\hat{A}}(j) \mathbb{I}^{(\text{row})}_j \]

- \( \mathbb{I}^{(\text{col})}_i = (E_{j \sim \mathcal{N}(i)} [H^{(l-1)} [j,:]])^T \frac{\partial J}{\partial E^{(l)}[i,:]} \)
- \( \mathbb{I}^{(\text{row})}_j = (H^{(l-1)} [j,:])^T \mathbb{E}_{i \sim \mathcal{N}(j)} \left[ \frac{\partial J}{\partial E^{(l)}[i,:]} \right] \)
- \( j \sim \mathcal{N}(i) \): Sampling node \( j \) from neighborhood of node \( i \) in \( \hat{A} \)
  - Sampling probability is proportional to \( \hat{A}[i,j] \)

\[ d_{\hat{A}}(i) = \text{sum } \hat{A}^T \]

\[ \mathbb{I}^{(\text{col})}_i = \mathbb{E} \]

\[ (H^{(l-1)})^T \]
Source of Unfairness: Row-wise Influence

- \( \frac{\partial J}{\partial W^{(l)}} \) is a linear summation of node influence weighted by its degree in \( \hat{A} \)

\[
\frac{\partial J}{\partial W^{(l)}} = \sum_{i=1}^{n} d_{\hat{A}}(i) I_{i}^{(\text{col})} = \sum_{j=1}^{n} d_{\hat{A}}(j) I_{j}^{(\text{row})}
\]

- \( I_{i}^{(\text{col})} = (E_{j \sim \mathcal{N}(i)} [H^{(l-1)}[j, :]])^T \frac{\partial J}{\partial E^{(l)}[i,:]} \)
- \( I_{j}^{(\text{row})} = (H^{(l-1)}[j,:])^T E_{i \sim \mathcal{N}(j)} \left[ \frac{\partial J}{\partial E^{(l)}[i,:]} \right] \)
- \( j \sim \mathcal{N}(i) \): Sampling node \( j \) from neighborhood of node \( i \) in \( \hat{A} \)
  - Sampling probability is proportional to \( \hat{A}[i,j] \)

\[
d_{\hat{A}}(j) = \text{sum}_{\hat{A}^T}
\]

\[
E \left[ \frac{\partial J}{\partial E^{(l)}} \right] = (H^{(l-1)})^T
\]
Source of Unfairness: Summary

• Gradient of loss w.r.t. weight
  \[
  \frac{\partial J}{\partial W^{(l)}} = \sum_{i=1}^{n} d_A(i)\Pi_i^{(col)} = \sum_{j=1}^{n} d_A(j)\Pi_j^{(row)}
  \]

• Intuitions
  - \(\Pi_i^{(col)}\) and \(\Pi_j^{(row)}\) → The directions for gradient descent
  - \(d_A(i)\) and \(d_A(j)\) → The importance of the direction

• High degree → more focus on the corresponding direction

• Question: Why does the node degree vary in \(\hat{A}\)?

Toy graph with adjacency matrix \(A\)

Node degree in \(A\)
- \(d_A(1) = 4\)
- \(d_A(2) = 2\)
- \(d_A(3) = 3\)
- \(d_A(4) = 2\)
- \(d_A(5) = 1\)

Node degree in \(\hat{A}\)
- \(d_{\hat{A}}(1) = 1.26\)
- \(d_{\hat{A}}(2) = 0.88\)
- \(d_{\hat{A}}(3) = 1.05\)
- \(d_{\hat{A}}(4) = 0.88\)
- \(d_{\hat{A}}(5) = 0.82\)

Different node degrees
Symmetric Normalization

- **Key idea**: Normalize the largest eigenvalue, but not degree.

- **Observation**: High degree in $A$ → high degree in $\hat{A}$
  - $\frac{\partial J}{\partial W(l)}$ favors high-degree nodes in $A$ due to such positive correlation.

- **Consequence**: $\frac{\partial J}{\partial W(l)}$ calculated using $\hat{A}$ is biased.

- **Example**
  
  Node $a$: $d_a(a) = 2$
  Node $b$: $d_a(b) = 1$

  - **Fair direction**: $\Pi_b^{(col)}$
  - **Biased direction**: Favor node $a$ by being closer to $\Pi_a^{(col)}$

  Node degree takes no effect
  Node degree is considered
Doubly Stochastic Matrix Computation

• How to mitigate unfairness in $\frac{\partial J}{\partial W^{(t)}}$?
  – **Intuition:** Enforce row sum and column sum of $\hat{A}$ to be 1
  – **Solution:** Doubly stochastic normalization on $\hat{A}$

• **Method:** Sinkhorn-Knopp algorithm
  – **Key idea:** Iteratively normalize the row and column of a matrix
  – **Complexity:** Linear time and space complexity
  – **Convergence:** Always converge iff. the matrix has total support

• **Question:** Can we find the doubly stochastic form of $\hat{A}$?
Existence of Doubly Stochastic Matrix

• Given
  – An undirected graph $\mathcal{G} = (A, X)$
  – The degree matrix $\tilde{D}$ of $A + I$
  – The renormalized graph Laplacian $\hat{A} = \tilde{D}^{-\frac{1}{2}}(A + I)\tilde{D}^{-\frac{1}{2}}$

• The Sinkhorn-Knopp algorithm always finds the unique doubly stochastic form $\hat{A}_{DS}$ of $\hat{A}$
  – (Check detailed proof in the paper)
Roadmap

• Motivation

• Theory: Source of Unfairness

• Algorithms: RawlsGCN

• Experiments

• Conclusion
The Family of RawlsGCN

• Gradient computation

\[
\left( \frac{\partial J}{\partial W^{(l)}} \right)_{\text{fair}} = \left( H^{(l-1)} \right)^T \hat{A}_{DS}^T \frac{\partial J}{\partial E^{(l)}}
\]

– Key term: \( \hat{A}_{DS} \) – Doubly-stochastic normalization of \( \hat{A} \)

• Proposed methods

– RawlsGCN-Graph: During data pre-processing, compute \( \hat{A}_{DS} \) and treat it as the input of GCN

– RawlsGCN-Grad: During optimization (in-processing), treat \( \hat{A}_{DS} \) as a normalizer to equalize the importance of node influence
RawlsGCN-Graph: Pre-processing

• **Intuition:** Normalize the input renormalized graph Laplacian into a doubly stochastic matrix

• **Key steps**
  1. Precompute the renormalized graph Laplacian $\hat{A}$
  2. Precompute $\hat{A}_{DS}$ by applying the Sinkhorn-Knopp algorithm
  3. Input $\hat{A}_{DS}$ and $X$ (node features) to GCN for training
RawlsGCN-Grad: In-processing

- **Intuition:** Equalize the importance of node influence in gradient computation

- **Key steps**
  1. Precompute the renormalized graph Laplacian $\tilde{A}$
  2. Input $\tilde{A}$ and $X$ (node features) to GCN
  3. Compute $\tilde{A}_{DS}$ by applying the Sinkhorn-Knopp algorithm
  4. Repeat until maximum number of training epochs
     - Compute the fair gradient $\left(\frac{\partial j}{\partial W^{(l)}}\right)_{\text{fair}} = (H^{(l-1)})^T \tilde{A}_{DS}^T \frac{\partial j}{\partial E^{(l)}}$ using $\tilde{A}_{DS}$
     - Update $W^{(l)}$ by the fair gradient $\left(\frac{\partial j}{\partial W^{(l)}}\right)_{\text{fair}}$
Roadmap

• Motivation ✔️
• Theory: Source of Unfairness ✔️
• Algorithms: RawlsGCN ✔️
• Experiments
• Conclusion
Experiments: Settings

• **Task:** Semi-supervised node classification

• **Datasets**

| Name                | Nodes | Edges  | Features | Classes | Median Deg. |
|---------------------|-------|--------|----------|---------|-------------|
| Cora-ML             | 2,995 | 16,316 | 2,879    | 7       | 3           |
| Citeseer            | 3,327 | 9,104  | 3,703    | 6       | 2           |
| Coauthor-CS         | 18,333| 163,788| 6,805    | 15      | 6           |
| Coauthor-Physics    | 34,493| 495,924| 8,415    | 5       | 10          |
| Amazon-Computers    | 13,752| 491,722| 767      | 10      | 22          |
| Amazon-Photo        | 7,650 | 238,162| 745      | 8       | 22          |

• **Baseline methods**
  – **Vanilla model:** GCN
  – **Fairness-aware models:** DEMO-Net, DSGCN, Tail-GNN, Adversarial Fair GCN, REDRESS

• **Metrics**
  – **Utility:** Classification Accuracy
  – **Bias:** Variance of average loss values
### Experiments: Node Classification

#### Observations
- RawlsGCN achieves the smallest bias
- Classification accuracy can be improved
  - mitigating the bias → higher accuracy for low-degree nodes
  - Higher overall accuracy

| Method             | Coauthor-Physics |          | Amazon-Computers |          | Amazon-Photo |          |
|--------------------|------------------|----------|------------------|----------|-------------|----------|
|                    | Acc.             | Bias     | Acc.             | Bias     | Acc.        | Bias     |
| GCN                | 93.96 ± 0.367    | 0.023 ± 0.001 | 64.84 ± 0.641    | 0.353 ± 0.026 | 79.58 ± 1.507 | 0.646 ± 0.038 |
| DEMO-Net           | 77.50 ± 0.566    | 0.084 ± 0.010 | 26.48 ± 3.455    | 0.456 ± 0.021 | 39.92 ± 1.242 | 0.243 ± 0.013 |
| DSGCN              | 79.08 ± 1.533    | 0.262 ± 0.075 | 27.68 ± 1.663    | 1.407 ± 0.685 | 26.76 ± 3.387 | 0.921 ± 0.805 |
| Tail-GNN           | OOM              |          | 76.24 ± 1.491    | 1.547 ± 0.670 | 86.00 ± 2.715 | 0.471 ± 0.264 |
| AdvFair            | 87.44 ± 1.132    | 0.892 ± 0.502 | 53.50 ± 5.362    | 4.395 ± 1.102 | 75.80 ± 3.563 | 51.24 ± 39.94 |
| REDRESS            | 94.48 ± 0.172    | 0.019 ± 0.001 | 80.36 ± 0.206    | 0.455 ± 0.032 | 89.00 ± 0.369 | 0.186 ± 0.030 |
| RawlsGCN-Graph (Ours) | 94.06 ± 0.196    | 0.016 ± 0.000 | 80.16 ± 0.859    | 0.121 ± 0.010 | 88.58 ± 1.116 | 0.071 ± 0.006 |
| RawlsGCN-Grad (Ours) | 94.18 ± 0.306    | 0.021 ± 0.002 | 74.18 ± 2.530    | 0.195 ± 0.029 | 83.70 ± 0.672 | 0.186 ± 0.068 |
Experiments: Node Classification

- **Observation:** RawlsGCN achieves more balanced loss and classification accuracy
  - Flatter slope of the regression line for RawlsGCN (in orange) than GCN (in blue)
Experiments: Efficiency

| Method                    | # Param. | Memory | Training Time |
|---------------------------|----------|--------|---------------|
| GCN (100 epochs)          | 48, 264  | 1,461  | 13.335        |
| GCN (200 epochs)          | 48, 264  | 1,461  | 28.727        |
| DEMO-Net                  | 11,999,880 | 1,661  | 9158.5        |
| DSGCN                     | 181,096  | 2,431  | 2714.8        |
| Tail-GNN                  | 2,845,567 | 2,081  | 94.058        |
| AdvFair                   | 89,280   | 1,519  | 148.11        |
| REDRESS                   | 48,264   | 1,481  | 291.69        |
| RawlsGCN-Graph (Ours)     | 48,264   | 1,461  | 11.783        |
| RawlsGCN-Grad (Ours)      | 48,264   | 1,461  | 12.924        |

- **Observation:** RawlsGCN has the best efficiency compared with other baseline methods
  - Same number of parameters and memory usage (in MB)
  - Much shorter training time (in seconds)
### Experiments: Ablation Study

| Method             | Normalization     | Acc.      | Bias      |
|--------------------|-------------------|-----------|-----------|
| RAWLSGCN-Graph     | Row               | 87.98 ± 0.791 | 0.076 ± 0.006 |
|                    | Column            | 88.32 ± 2.315 | 0.138 ± 0.112 |
|                    | Symmetric         | 89.12 ± 0.945 | 0.071 ± 0.005 |
| Doubly Stochastic  |                   | 88.58 ± 1.116 | **0.071 ± 0.006** |
| RAWLSGCN-Grad      | Row               | 82.86 ± 1.139 | 0.852 ± 0.557 |
|                    | Column            | 84.96 ± 1.235 | 0.221 ± 0.064 |
|                    | Symmetric         | 82.92 ± 1.121 | 0.744 ± 0.153 |
| Doubly Stochastic  |                   | 83.70 ± 0.672 | **0.186 ± 0.068** |

**Observation:** Doubly stochastic normalization is the best normalization technique to balance accuracy and fairness.
Roadmap

• Motivation
• Theory: Source of Unfairness
• Algorithms: RawlsGCN
• Experiments
• Conclusion
Conclusion

• **Problem:** Enforce the Rawlsian difference principle on GCN

• **Source of unfairness**
  – Analysis on the gradient w.r.t. model weights
  – Doubly stochastic normalization on the graph

• **Solution:** RawlsGCN
  – Pre-processing by RawlsGCN-Graph
  – In-processing by RawlsGCN-Grad

• **Results**
  – Effectiveness in bias mitigation while maintaining accuracy
  – Significant improvement in efficiency

• **More details in the paper**
  – Proofs and analysis
  – Detailed experiments