# Adversarial Learning Data Augmentation for Graph Contrastive Learning in Recommendation

Junjie Huang, Qi Cao, Ruobing Xie, Shaoliang Zhang,
Feng Xia, Huawei Shen, Xueqi Cheng

Institute of Computing Technology, Chinese Academy of Sciences
Tencent WeChat

# Table of Contents

# Outline

# Introduction: Recommender Systems & GNNs

**Context**

- Collaborative Filtering (CF) models user-item interactions.
- Modeled as Bipartite Graph $G = (\mathcal{U}, \mathcal{I}, \mathcal{E})$.
- Graph Neural Networks (GNNs) capture high-order structure.
- Examples: NGCF, LightGCN.

# Introduction: Recommender Systems & GNNs

**Context**

- Collaborative Filtering (CF) models user-item interactions.
- Modeled as Bipartite Graph $G = (\mathcal{U}, \mathcal{I}, \mathcal{E})$.
- Graph Neural Networks (GNNs) capture high-order structure.
- Examples: NGCF, LightGCN.

**The Problem**

- **Data Sparsity**: Limited interactions for learning quality embeddings.
- Performance degrades significantly on sparse datasets.

# Solution: Graph Contrastive Learning (GCL)

**Current Approach: GCL**

- Self-supervised learning paradigm.
- **Data Augmentation**: Create multiple views of the graph (e.g., edge dropping).
- **Contrastive Loss**: Maximize agreement between views (InfoNCE).

# Solution: Graph Contrastive Learning (GCL)

**Current Approach: GCL**

- Self-supervised learning paradigm.
- **Data Augmentation**: Create multiple views of the graph (e.g., edge dropping).
- **Contrastive Loss**: Maximize agreement between views (InfoNCE).

**Limitations of Existing GCL**

- Relies on **heuristic** augmentation (random sampling).
- Random dropping may destroy structural information.
- Improper augmentation hinders performance.
- **Goal**: Learn an *optimal* augmentation strategy automatically.

# Outline

# Background: Principles of Contrastive Learning

**InfoMax Principle** (Standard GCL)

- Objective: Maximize Mutual Information (MI) between views.
- Learn representations invariant to perturbations.
- Used by most current methods (e.g., SGL, SimGCL).

# Background: Principles of Contrastive Learning

**InfoMax Principle** (Standard GCL)

- Objective: Maximize Mutual Information (MI) between views.
- Learn representations invariant to perturbations.
- Used by most current methods (e.g., SGL, SimGCL).

**InfoMin Principle** (Our Inspiration)

- Concept: A good set of views shares the **minimal** information necessary.
- Removing redundancy forces the model to learn robust features.
- Stronger, harder augmentations lead to better downstream performance.

# Outline

# Overview of LDA-GCL

**Proposed Framework: LDA-GCL**

- **L**earnable **D**ata **A**ugmentation for **G**raph **C**ontrastive **L**earning.
- Combines InfoMax and InfoMin principles via adversarial training.

# Overview of LDA-GCL

**Proposed Framework: LDA-GCL**

- **L**earnable **D**ata **A**ugmentation for **G**raph **C**ontrastive **L**earning.
- Combines InfoMax and InfoMin principles via adversarial training.

**Core Components**

1. **Edge Operating**: New augmentation strategy (Adding + Dropping).
2. **Learnable View Generator ($t$)**: Learns to create "hard" views.
3. **GNN Encoder ($f$)**: Learns robust embeddings.
4. **Adversarial Optimization**: Min-Max game.

# Framework Architecture

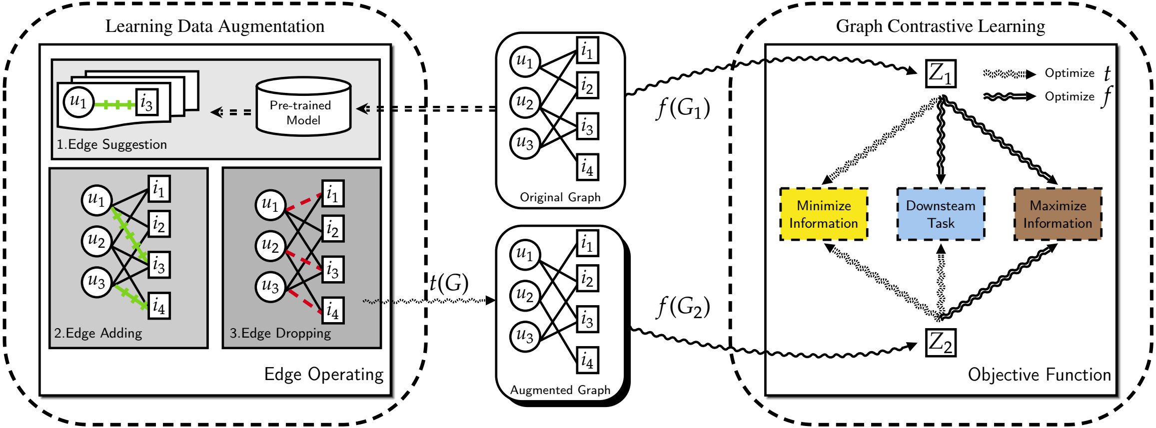


**Figure:** LDA-GCL Framework: Learning Data Augmentation + GCL.

# 1. Edge Operating Augmentation

**Challenge with Random Augmentation**

- Randomly adding edges introduces noise.
- Sampling from all non-observed edges is $O(|\mathcal{V}|^2)$ (too expensive).

# 1. Edge Operating Augmentation

**Challenge with Random Augmentation**

- Randomly adding edges introduces noise.
- Sampling from all non-observed edges is $O(|\mathcal{V}|^2)$ (too expensive).

**Proposed Solution: Pre-trained Candidates**

- Train a light GNN (e.g., LightGCN) first.
- Identify top-$K_u$ likely items for user $u$.
- **Edge Candidates** ($\mathcal{E}_{cand}$): Original Edges $\cup$ Suggested Edges.

# 2. Learnable Edge Operator

**Generating the View**

- Instead of random sampling, learn a probability $p_{u,i}$ for each candidate edge.
- Use an MLP edge operator model $t$:

$$\omega_{u,i} = \mathrm{MLP}([z_u \odot z_i] \parallel \mathbb{1}_{\mathcal{E}}(e_{u,i}))$$

- $z_u, z_i$: Embeddings.
- $\mathbb{1}_{\mathcal{E}}$: Indicator (is original edge or added?).

# 2. Learnable Edge Operator

## Generating the View

- Instead of random sampling, learn a probability $p_{u,i}$ for each candidate edge.
- Use an MLP edge operator model $t$:

$$\omega_{u,i} = \mathrm{MLP}([z_u \odot z_i] \parallel \mathbb{1}_{\mathcal{E}}(e_{u,i}))$$

- $z_u, z_i$: Embeddings.
- $\mathbb{1}_{\mathcal{E}}$: Indicator (is original edge or added?).

## Differentiable Sampling (Gumbel-Max)

- To allow backpropagation through sampling:

$$p_{u,i} = \mathrm{sigmoid}\left(\frac{\log \delta - \log(1-\delta) + \omega_{u,i}}{\tau}\right)$$

- Generates augmented adjacency matrix $\mathbf{A}'$.

# 3. Adversarial Objective (The Min-Max Game)

**Objective Function**

- Optimization involves the Augmenter $t$ and Encoder $f$:

$$\min_{t} \quad \lambda_t I(f(G); f(t(G))) + \mathcal{L}(f(t(G)), y)$$

$$\max_{f} \quad I(f(G); f(t(G))) - \mathcal{L}(f(G), y)$$

# 3. Adversarial Objective (The Min-Max Game)

**Objective Function**

- Optimization involves the Augmenter $t$ and Encoder $f$:

$$\min_{t} \quad \lambda_t I(f(G); f(t(G))) + \mathcal{L}(f(t(G)), y)$$
$$\max_{f} \quad I(f(G); f(t(G))) - \mathcal{L}(f(G), y)$$

- **Minimizing $I$ (for $t$)**: InfoMin. Create views that share *minimal* info (hard samples).
- **Maximizing $I$ (for $f$)**: InfoMax. Encoder tries to align views despite augmentation.

# Loss Functions: Concrete Form

**Fix Augmenter $t$, Optimize Encoder $f$**

- Standard GCL training step.
- Maximize agreement (InfoNCE) + Minimize BPR loss.

$$\mathcal{L}_f = \mathcal{L}_{\text{BPR}}(f(G)) + \lambda_{ssl}\mathcal{L}_{\text{NCE}}(f(G), f(t(G)))$$

# Loss Functions: Concrete Form

**Fix Augmenter $t$, Optimize Encoder $f$**

- Standard GCL training step.
- Maximize agreement (InfoNCE) + Minimize BPR loss.

$$\mathcal{L}_f = \mathcal{L}_{\text{BPR}}(f(G)) + \lambda_{ssl}\mathcal{L}_{\text{NCE}}(f(G), f(t(G)))$$

**Fix Encoder $f$, Optimize Augmenter $t$**

- Adversarial step.
- *Minimize* agreement (make the view different).
- Maintain task performance (don't break the graph).

$$\mathcal{L}_t = \mathcal{L}_{\text{BPR}}(f(t(G))) - \lambda_2\mathcal{L}_{\text{NCE}}(f(G), f(t(G)))$$

# Training Algorithm

**Input:** Graph $G$, Pre-trained $f_0$
Generate candidate edges using $f_0$;
Initialize $t$ and $f$;
**for** *epoch* $=$ *1* **to** *T* **do**
    **for** *each mini-batch* **do**
        `/* Optimize Augmenter (InfoMin) */`
        Freeze $f$, Unfreeze $t$;
        Generate view $t(G)$;
        Update $t$ via $\mathcal{L}_t$ (Minimize MI);
        `/* Optimize Encoder (InfoMax) */`
        Freeze $t$, Unfreeze $f$;
        Generate view $t(G)$;
        Update $f$ via $\mathcal{L}_f$ (Maximize MI);
    **end**
**end**

**Algorithm 1:** LDA-GCL Adversarial Training

# Outline

# Experimental Setup

**Datasets**

- Yelp, Gowalla, Amazon-Book, Alibaba-iFashion.
- Variety in size and density (Alibaba is very sparse).

**Baselines**

- **MF**: BPRMF, NeuMF.
- **GNN**: NGCF, DGCF, LightGCN.
- **GCL**: SGL (Edge Drop), SimGCL (Noise), NCL (Neighbor Contrast).

**Metrics**

- Recall@10, 20, 50.
- NDCG@10, 20, 50.

# Performance Comparison (Main Results)

- LDA-GCL consistently outperforms baselines.
- Significant improvement over SGL and NCL.

| Dataset | LightGCN | SGL | NCL | **LDA-GCL** |
|---|---|---|---|---|
| **Yelp** (Recall@20) | 0.1001 | 0.1072 | 0.1135 | **0.1190** |
| **Amazon** (Recall@20) | 0.1210 | 0.1281 | 0.1395 | **0.1456** |
| **Gowalla** (Recall@20) | 0.1969 | 0.1969 | 0.2131 | **0.2144** |
| **Alibaba** (Recall@20) | 0.0612 | 0.0774 | 0.0729 | **0.0882** |

**Table:** Subset of experimental results (Recall@20).

# Performance Comparison (Main Results)

- LDA-GCL consistently outperforms baselines.
- Significant improvement over SGL and NCL.

| Dataset | LightGCN | SGL | NCL | **LDA-GCL** |
|---|---|---|---|---|
| **Yelp** (Recall@20) | 0.1001 | 0.1072 | 0.1135 | **0.1190** |
| **Amazon** (Recall@20) | 0.1210 | 0.1281 | 0.1395 | **0.1456** |
| **Gowalla** (Recall@20) | 0.1969 | 0.1969 | 0.2131 | **0.2144** |
| **Alibaba** (Recall@20) | 0.0612 | 0.0774 | 0.0729 | **0.0882** |

**Table:** Subset of experimental results (Recall@20).

- Highest gains on **Alibaba-iFashion** (sparsest dataset).
- Demonstrates effectiveness in handling data sparsity.

# Analysis: Why it works? (Sparsity)

**Performance by User Group (Interaction Frequency)**

[Fig: Gowalla Groups] [Fig: Alibaba Groups]

**Figure:** LDA-GCL vs SGL vs LightGCN on user groups.

- LDA-GCL outperforms LightGCN and SGL across all groups.
- Gains are largest for sparse users (low interaction count).

# Ablation Study

**Components Analysis**

- **w/o EA**: Removing Edge Adding (only learnable dropping).
  - Performance drops → Adding edges helps explore structure.
- **w NGCF**: Using NGCF for edge candidates instead of LightGCN.
  - Performance drops → Quality of candidates matters.
- **DA-GCL**: Random adding/dropping (No learning).
  - LDA-GCL is superior → **Learnable** augmentation is key.

# Ablation Study

**Components Analysis**

- **w/o EA**: Removing Edge Adding (only learnable dropping).
  - Performance drops $\rightarrow$ Adding edges helps explore structure.
- **w NGCF**: Using NGCF for edge candidates instead of LightGCN.
  - Performance drops $\rightarrow$ Quality of candidates matters.
- **DA-GCL**: Random adding/dropping (No learning).
  - LDA-GCL is superior $\rightarrow$ **Learnable** augmentation is key.

**Conclusion from Ablation**

- Both Edge Operating (Adding) and Learnable Selection (Adversarial) are crucial.

# Parameter Analysis

**Impact of Adversarial Weight $\lambda_t$**

- $\lambda_t$ controls the strength of minimizing Mutual Information.
- $\lambda_t = 0$: No adversarial training.

**[Fig: Parameter Sensitivity]**

- Results show $\lambda_t > 0$ yields better performance.
- Confirms the **InfoMin** principle improves robustness.

# Outline

# Conclusion

## Summary

- Proposed **LDA-GCL**: A theoretical framework for GCL in Recommendation.
- Integrated **InfoMin** (Adversarial View Gen.) and **InfoMax** (Contrastive).
- Designed **Edge Operating** augmentation (Adding + Dropping).

# Conclusion

**Summary**

- Proposed **LDA-GCL**: A theoretical framework for GCL in Recommendation.
- Integrated **InfoMin** (Adversarial View Gen.) and **InfoMax** (Contrastive).
- Designed **Edge Operating** augmentation (Adding + Dropping).

**Key Takeaways**

- Replaces heuristic augmentation with learned strategies.
- Effectively handles data sparsity.
- Achieves state-of-the-art performance on benchmarks.

**Future Work**

- Improve training efficiency (reduce complexity of learning $t$).

# Reference I