Knowledge Graph-Based Recommendation

Presented By:

Agnik Saha

Introduction to Knowledge Graphs

What is a Knowledge Graph (KG)?

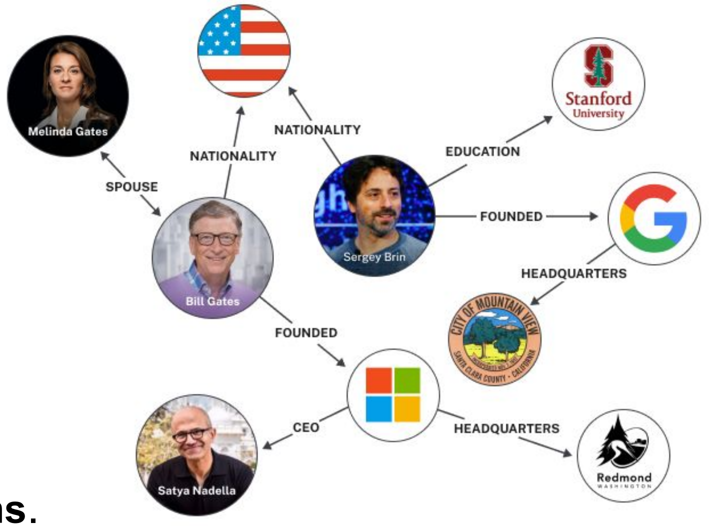
Knowledge graphs (KGs) store structured knowledge as a collection of triples

$KG = \{(h, r, t) \subseteq E \times R \times E\}$,

where E = {set of entities}

and R = {set of relations}

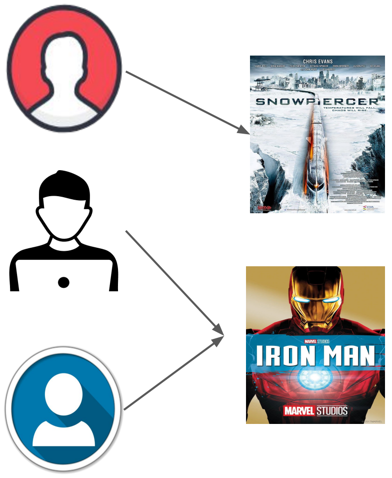
Example: Microsoft → CEO → Satya Nadella



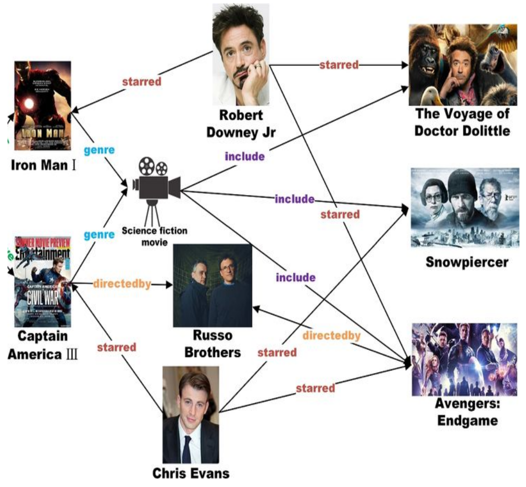
Why Use Knowledge Graphs in Recommendations?

- **Captures structured relationships** among items.
- Helps in **personalized, explainable recommendations**.
- Example: If a user likes **Sci-Fi movies**, the KG can recommend movies with **similar directors or actors.**

Problem



User-item Interaction Graph $\mathcal{G}_u$



Knowledge Graph $\mathcal{G}_k$

Task: Learn a recommender model $f(u,i \mid \mathcal{G}_u, \mathcal{G}_k, \boldsymbol{\theta})$ that outputs the likelihood of user u interacting with item i

Challenges:

- Limited Interpretability: Difficulty in understanding the reasoning behind recommendations.
- Reduced Accuracy: Noise and bias introduced by noisy KG relations and spurious correlations.
- Sparsity Issues: Insufficient data for effective learning and recommendations.
- Struggle with cold-start users (with few interactions) and long-tail items (rarely interacted)

Existing Solutions

Graph Neural Networks (GNNs)
Model high-order relationships in KGs.

Path-Based Method
Use semantic reasoning paths for interpretability.

Causal Inference
Reduce biases with causal graphs and counterfactual reasoning.

Self-Supervised Learning
Mask and reconstruct KG relations to manage noise.

Embedding-Based Methods
Learn vector representations of KG entities and relations.

Hybrid Models
Combine collaborative filtering with KG embeddings for context

Attention Mechanisms
Highlight key entities and relations for explainability

Knowledge-Enhanced Pre-training
Pretrain models on KG data for better initialization

Introduction

KGCR: Knowledge Graph-based Causal Recommendation [1]

- Improves collaborative filtering using causal inference.
- Incorporates counterfactual reasoning to address biases.

KGRec: Knowledge Graph Self-Supervised Rationalization [2]

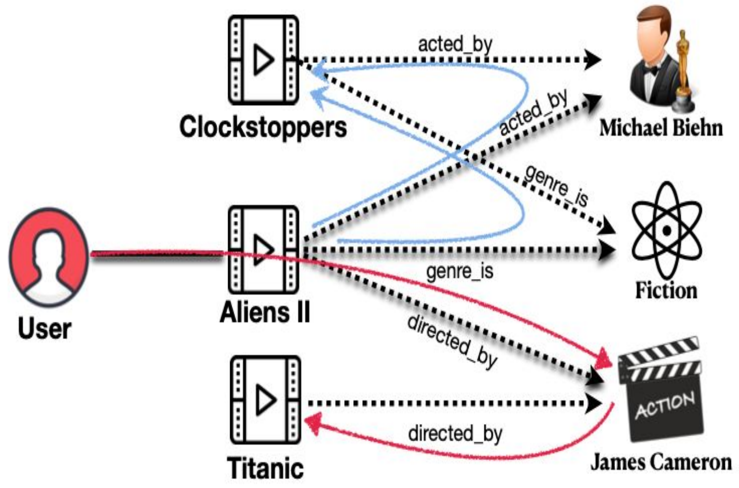
- Identifies important KG triplets by masking irrelevant connections.
- Enhances recommendations with rational reconstruction and contrastive learning.

[1] Wei, Y., Wang, X., Nie, L., Li, S., Wang, D., & Chua, T. S. (2022). Causal inference for knowledge graph based recommendation. IEEE Transactions on Knowledge and Data Engineering, 35(11), 11153-11164.
[2] Yang, Y., Huang, C., Xia, L., & Huang, C. (2023, August). Knowledge graph self-supervised rationalization for recommendation. In Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining (pp. 3046-3056).

Causal Inference For Knowledge Graph Based Recommendation

Motivation (KGCR)

A user's preference for an item may appear to depend on attributes (e.g., genre, director) due to hidden correlations introduced by the KG structure (confounders).



KG-based recommendation

Observed Bias !!!

- A user prefers movies directed by **James Cameron**.
- Due to the KG structure, the movie **Clockstoppers** (with the same actor as another favorite movie, **Aliens II**) may score **higher** than **Titanic** (directed by James Cameron).
- This happens because the **actor connection** provides a **shortcut**, bypassing the user's true preference for **directors**.

Methodology

1 Constructing the Causal Graph

Define **users, items, and attributes** as nodes

Identify **confounder (K)** that introduces bias in user preference learning.

2 Learning User & Item Representations

The user preference representation is modeled using Graph Convolutional Networks (**GCNs**) on **user-attribute** and **item-attribute** bipartite graphs

Message passing over the graph incorporates structural information **while maintaining fine-grained attribute-level representations**

3 Counterfactual Inference to Remove Bias

Apply **do-calculus** to eliminate spurious correlations.

Estimate **user preferences independently of item structures**

4 Compute Debiased Similarity Scores

Decompose similarity into **Total Effect (TE) & Natural Direct Effect (NDE)**

Subtract NDE from TE to remove bias

5 Model Optimization & Training

The **Bayesian Personalized Ranking (BPR) loss** is used to optimize user-item interactions, incorporating both **debiased similarity scores and collaborative filtering scores**

Introduction to Causal Inference

What is Causal Inference?

- Understanding cause-effect relationships rather than just correlations.
- Example:
  - **Correlation:** Users who watch Sci-Fi also watch Action movies.
  - **Causation:** Users prefer Sci-Fi movies **because of specific attributes (e.g., director, storyline)**.

Why Do We Need Causal Inference in KG-Based Recommendations?

- Traditional KG-based methods rely on correlation-based predictions.
- Causal methods remove bias by eliminating spurious correlations.

Solution: Use causal graphs and counterfactual inference to debias recommendations.

Causal Inference: Key Concepts

Counterfactuals: These hypothetical scenarios compare what happened with what would have happened under different conditions.

Causal Structure:

$$X \leftarrow M \rightarrow Y$$

- **X (Skills & Experience)** → Candidate's real competencies.
- **M (University Prestige)** → How prestigious the university is.
- **Y (Hiring Decision)** → Likelihood of getting the job.

Total Effect (TE): Total causal effect of both university prestige (M) and skills (X) on hiring decision (Y), including both direct and indirect pathways.
Direct Effect (DE): The effect of university prestige (M) on hiring (Y), holding skills (X) constant.
Total Indirect Effect (TIE): The effect of university prestige (M) on the hiring decision (Y) that operates through skills (X).

Key Concepts (do-Operator)

Without the do-Operator: Hiring decisions are influenced by both skills & prestige bias.

$$P(Y \mid X)$$

With the do-Operator: To measure the true effect of skills (X) on hiring (Y), we use:

$$P(Y \mid do(X))$$

This removes the confounding effect of university prestige and ensures hiring is analyzed independent of university bias (Z).

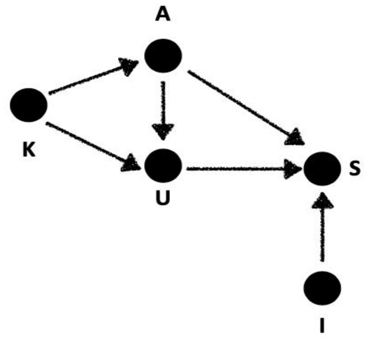
$$P(Y|do(X)) = \sum_{Z} P(Y|X,Z)P(Z)$$

Here, $P(Y \mid X,Z)$ = Probability of getting hired given skills and university prestige.

$P(Z)$ = Distribution of university prestige.

The Causal Graph

KGCR models user-item interactions using a causal graph.

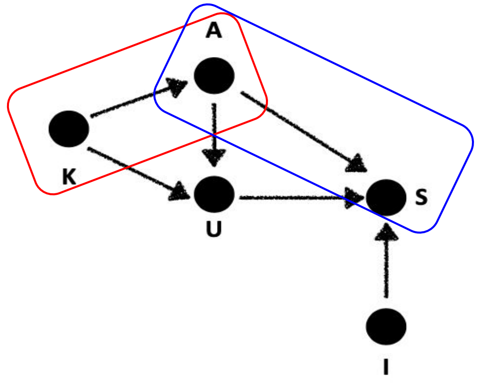


(a) Proposed Causal Graph

- U (User Preferences on attributes)
- I (Item Representation)
- A (User-Interacted Attributes)
- K (The relationships between items and attributes)
- S (Recommendation score)

- ***K → A**:* Structure information (K) **decides which attributes (A)** are exposed to the user.
- ***K → U**:* Structure information (K) also influences **user preference (U)** since users observe items through attributes.
- ***(A, K) → U**:* The **user preference (U) is affected** by both the **user's interacted attributes (A)** and the underlying structure information (K).
- ***(A, U, I) → S***: The **similarity score (S) depends** on user preference (U), item representation (I), and attributes (A).
- ***A → S*** (Bias Path): The **user's interacted attributes (A) directly influence the similarity score (S)**, introducing a bias.

The Causal Graph



(a) Proposed Causal Graph

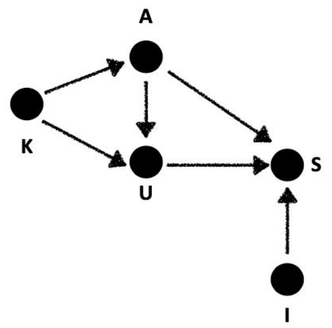
- U (User Preferences on attributes)
- I (Item Representation)
- A (User-Interacted Attributes)
- K (The relationships between items and attributes)
- S (Recommendation score)

Key Challenges Identified:

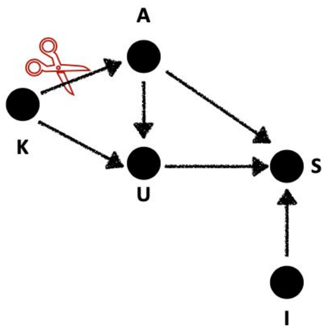
- Confounding Effect ($A \leftarrow K \rightarrow U$): The knowledge structure (K) influences which attributes (A) users are exposed to, creating bias in user preferences (Accurately modeling user preferences for **attributes** in KGs.)

- Bias Effect ($U \leftarrow A \rightarrow S$): Attributes (A) can create shortcuts in similarity scoring (S), leading to biased recommendations (Eliminating **bias** caused by structural **shortcuts** in KG.)

Confounding Effect



(a) Proposed Causal Graph



(b) Intervention on Causal Graph

The causal graph shows a **spurious correlation** where **K (Structural Information)** affects both:

1. **User preference (U) on attributes (A)**
2. **Exposure of attributes (A) to users**

This creates a **backdoor path**: $A \leftarrow K \rightarrow U$

Issue: The observed user preference on attributes may be biased by **structural constraints in the KG**.

Solution: To remove this bias, causal intervention (do-calculus) is applied to cut off $K \rightarrow A$.

Causal Intervention

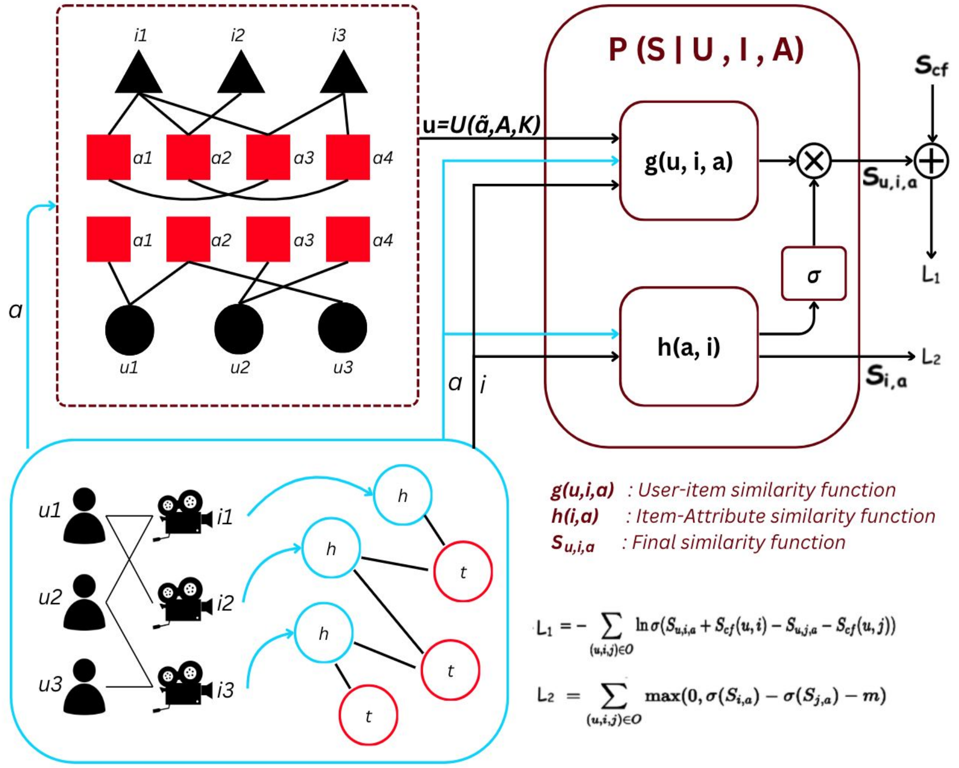
- The model applies **causal intervention on A** using **do-calculus** to break the backdoor path: A ← K → U
- By **intervening on A**, the user preference learning process **removes the confounding influence of K**.
- This intervention is mathematically expressed as:

$$P(S|do(A=a)) = \sum_{k \in K} P(S|U(a,k), I=i, a)P(k)$$

- *S: Similarity score between user and item.*
- *A: Attributes being intervened upon.*
- *K: Confounding variables.*
- *P(k): Distribution over confounders.*
- *do(A=a): Forces A to a fixed value, eliminating confounders' influence.*

Proposed Model

- Constructs **User-Item** and **Item-Attribute** bipartite graphs.
- Applies **graph convolutional networks (GCN)** to refine user, item, and attribute embeddings.
- Removes bias from KG structure for better preference modeling.
- Computes **user-item similarity** g(u,i,a) and **item-attribute similarity** h(i,a).
- The final recommendation score is a sum of collaborative filtering (CF) signals and debiased knowledge-based similarity scores.



g(u,i,a) : User-item similarity function
h(i,a) : Item-Attribute similarity function
$S_{u,i,a}$: Final similarity function

$$L_1 = -\sum_{(u,i,j)\in O} \ln \sigma(S_{u,i,a} + S_{cf}(u,i) - S_{u,j,a} - S_{cf}(u,j))$$

$$L_2 = \sum_{(u,i,j)\in O} \max(0, \sigma(S_{i,a}) - \sigma(S_{j,a}) - m)$$

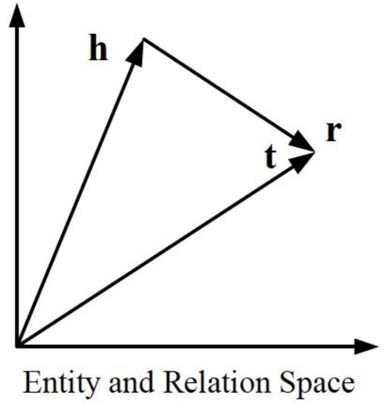
Knowledge Graph Embedding

Compute embeddings E of nodes and edges in the knowledge graph as:

$$L_{KG} = \sum_{(h,r,t)\in T} \sum_{(h',r,t')\in T'} [\gamma + d(e_h + e_r, e_t) - d(e'_h + e_r, e'_t)]_+$$

where:

- T = true triples.
- T' = corrupted triples.
- $d()$ = Euclidean distance.
- γ = margin hyperparameter.



Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, *26*.

Model Implementation

User and Attribute Representation:

- Construct bipartite graphs (user-attribute and item-attribute).
- Perform Graph Convolutional Networks (GCN) to aggregate local structure information:

$$p^{(l+1)} = \sum_{q \in N(p)} \frac{1}{\sqrt{|N(p)||N(q)|}} q^{(l)}$$

Where *N(p)* denotes neighbors and |N(p)| and |N(q)| represent number of neighbors of node p and q.

Scoring Function:

- Compute user-item similarity: $S_{u,i,a} = f(u,i,a) = g(u,i) \cdot \sigma(h(i,a))$
- where:
  - g(u,i) models the similarity score comes from the user preference to item
  - h(i,a) models affinity between item and user-interacted attributes.

Loss Function Optimization

Optimize using combined loss functions:

1. Ranking-based loss for similarity scoring $$L_1 = -\sum_{(u,i,j)\in O} \ln \sigma(S_{u,i,a} + S_{cf}(u,i) - S_{u,j,a} - S_{cf}(u,j))$$

   $O = \{(u,i,j) \mid (u,i) \in O^+, (u,j) \in O^-\}$, O^+ denotes positive and O^- denotes negative user-item pairs

   $S_{u,i,a}$: similarity score computed.

   $S_{cf}(u,i)$: Collaborative filtering score derived from user-item interactions.

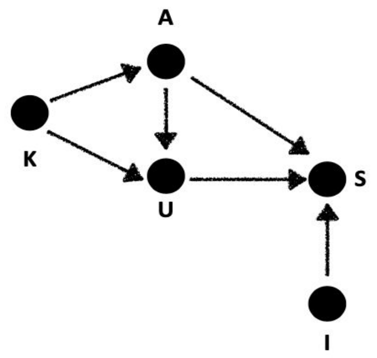
   $S_{u,j,a}$, $S_{cf}(u,j)$: Scores for negative samples (items not interacted with).

2. Margin loss for attribute similarity: $$L_2 = \sum_{(u,i,j)\in O} \max(0, \sigma(S_{i,a}) - \sigma(S_{j,a}) - m)$$

Final objective:

$$L = L_1 + \alpha L_2 + ||\Theta||^2$$

Counterfactual Inference



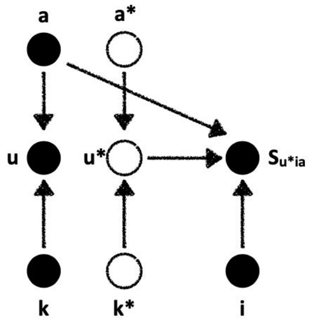
(a) Proposed Causal Graph

Challenge:

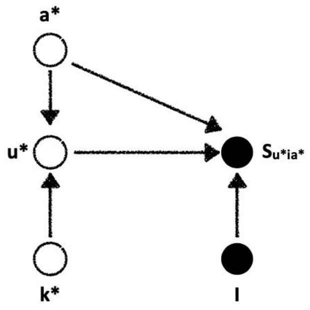
- The direct effect of A on S provides shortcut correlations, causing biased similarity scores and overestimation of certain items.

- Total effect (TE): Overall influence of user preference (U) and attributes (A) on similarity score (S).
- Natural Direct Effect (NDE): the direct impact of attributes (A) on similarity score (S) while holding the user preference (U) constant.
- Total Indirect Effect (TIE): effect of confounders indirectly propagating through the causal path.

Counterfactual Inference



(a) Counterfactual World



(b) Reference Situation

Removes confounder influence

Influenced by attributes and confounders

$$TE = S_{u,i,a} - S_{u^*,i,a^*}$$

Includes both direct and indirect effects (via confounders).

$S_{u,i,a}$: Observed score with the actual attributes.
S_{u^*, i, a^*}: Counterfactual score when attributes are **controlled** to eliminate biases.

$$NDE = S_{u^*,i,a} - S_{u^*,i,a^*}$$

Direct impact of attributes while keeping confounders fixed.

$S_{u^*,i,a}$: Counterfactual score with actual attributes.
S_{u^*, i, a^*}: Counterfactual score after controlling confounders.

$$TIE = TE - NDE$$

tells us how much of the bias is due to confounders

Counterfactual Inference

By removing the Natural Direct Effect (NDE) from the Total Effect (TE), we obtain the **debiased effect** of the user and item on the score. This is termed as the **Total Indirect Effect (TIE)** and is formulated as:

$$
\begin{aligned}
TIE &= TE - NDE \\
&= S_{u,i,a} - S_{u^*,i,a^*} - S_{u^*,i,a} + S_{u^*,i,a^*} \\
&= g(\mathbf{u}, i)\sigma(h(i,a)) - g(\mathbf{u}^*, i)\sigma(h(i,a)) \\
&= (g(\mathbf{u}, i) - s_i)\sigma(h(i,a)).
\end{aligned}
$$

Since the reference value **u*** is independent of the specific user, **s$_i$** can be computed as the mean output of **g(u, i)**

Debiased Similarity score : $y_{ui} = S_{ui}^{cf} + S_{ui}^{kg} = \mathbf{u}_{cf}^T \cdot \mathbf{i}_{cf} + (g(\mathbf{u}, i) - s_i)\sigma(h(i,a))$

Algorithm

Input:

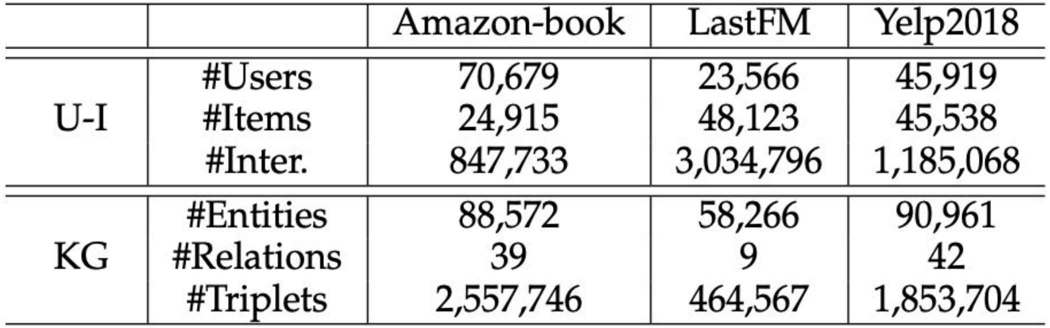
- **O+**: User-item interactions.
- $\mathbf{G_{kg}}$: Knowledge graph (triples).

Output:

- Optimized parameters **Θ** including user/item embeddings.
- Predictions for collaborative filtering and similarity scores ($\mathbf{S_{cf}}$, $\mathbf{S_{u,i,a}}$).

```
1. Initialize model parameters $\mathbf{\Theta}$,
2. Calculate embeddings for KG (TransE).
3. Initialize user-items, item-attribute and user-attribute distributions.
4. For each graph convolution layer:
    a. Compute user and item representations using GCN.
    b. Compute similarity scores:
        i. User-item similarity $\mathbf{g(u, i)}$.
        ii. Attribute similarity $\mathbf{h(i, a)}$.
5. Apply Causal Inference for Debiasing.
6. Optimize loss function combining collaborative filtering loss and KG-based scores, and apply regularization.
7. Apply Adam optimizer to update parameters.
8. Return optimized embeddings and scores.
```

Dataset

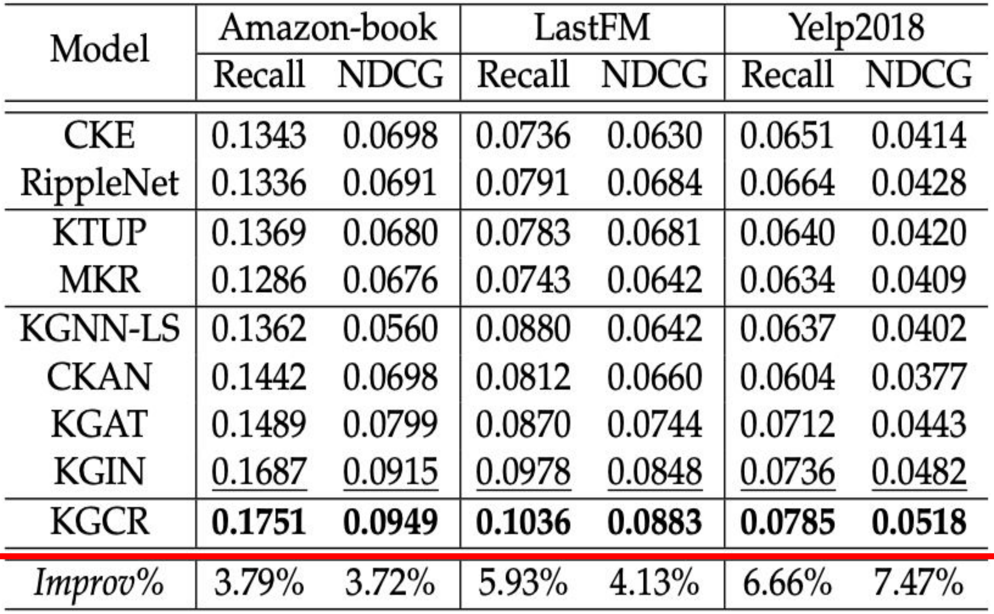


		Amazon-book	LastFM	Yelp2018
U-I	#Users	70,679	23,566	45,919
	#Items	24,915	48,123	45,538
	#Inter.	847,733	3,034,796	1,185,068
KG	#Entities	88,572	58,266	90,961
	#Relations	39	9	42
	#Triplets	2,557,746	464,567	1,853,704

Dataset of KGCR

- **Amazon-book**: *KGCR improves Recall@20 by **3.79%** over the strongest baseline.*
- **LastFM**: *KGCR achieves the **highest improvement (+5.93%)** among all datasets*
- **Yelp2018**: *KGCR outperforms all baselines with a **6.66% gain in Recall@20**.*

Experimental Result



Model	Amazon-book		LastFM		Yelp2018	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
CKE	0.1343	0.0698	0.0736	0.0630	0.0651	0.0414
RippleNet	0.1336	0.0691	0.0791	0.0684	0.0664	0.0428
KTUP	0.1369	0.0680	0.0783	0.0681	0.0640	0.0420
MKR	0.1286	0.0676	0.0743	0.0642	0.0634	0.0409
KGNN-LS	0.1362	0.0560	0.0880	0.0642	0.0637	0.0402
CKAN	0.1442	0.0698	0.0812	0.0660	0.0604	0.0377
KGAT	0.1489	0.0799	0.0870	0.0744	0.0712	0.0443
KGIN	0.1687	0.0915	0.0978	0.0848	0.0736	0.0482
KGCR	**0.1751**	**0.0949**	**0.1036**	**0.0883**	**0.0785**	**0.0518**
Improv%	3.79%	3.72%	5.93%	4.13%	6.66%	7.47%

Performance of KGCR

- **KGCR outperforms all baselines** across all datasets, confirming its effectiveness.
- **Performance improvements are significant**, particularly on LastFM and Yelp2018.
- **KGIN is the strongest baseline**, but KGCR surpasses it due to causal intervention and debiasing.

Ablation Study: Effect of Causal Components

Model Variant	Amazon-book (Recall@20)	LastFM (Recall@20)	Yelp2018 (Recall@20)
KGCR (Full Model)	**0.1751**	**0.1036**	**0.0786**
w/o Deconfounded User Preference (DC)	0.1565	0.0867	0.0721
w/o Counterfactual Inference (CI)	0.1716	0.0996	0.0774

- **Removing deconfounded user preference (DC) significantly reduces performance**, confirming that structure-aware causal modeling is crucial.
- **Counterfactual inference (CI)** also plays an important role, as removing it causes a slight drop in performance.
- Both causal intervention and debiasing are essential for achieving optimal recommendation accuracy.

Summary (KGCR)

- Introduces a **causal framework** to deconfound user preference learning and mitigate biases in similarity-based scoring.
- Uses **counterfactual inference** to remove bias from similarity scoring, improving **recommendation fairness and accuracy**.
- Demonstrates significant improvements in **Recall@20 and NDCG@20** on datasets like **Amazon-Book, LastFM, and Yelp2018**.

Knowledge Graph Self-Supervised Rationalization For Recommendation

Motivation (KGRec)

1. **Reducing Noise in Knowledge Graphs**

   Traditional KG-based models include all connections, leading to noisy triplets influencing predictions. KGRec filters irrelevant triplets to enhance relevance.

2. **Aligning KG Signals with User Preferences**

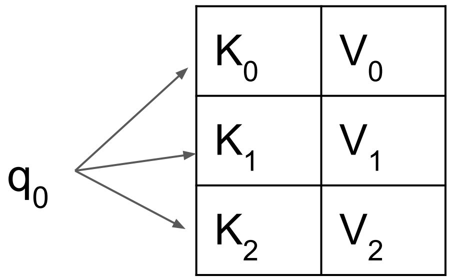
   Existing models fail to bridge KG signals and user-item interactions. KGRec enforces alignment using contrastive learning for better accuracy.

3. **Enhancing Representations with Self-Supervision**

   Standard KG embeddings struggle with generalization. KGRec applies masked autoencoding and contrastive learning for robust and interpretable representations.

Attention

Attention is a technique that allows models to focus on the most important parts of input data. It assigns different weights to different parts of the input so that the model can prioritize important relationships.



$$A(q, K, V) = \sum_i \frac{e^{s(q,k_i)}}{\sum_j e^{s(q,k_j)}} v_i$$

$$s(q,k) = \frac{q \cdot k}{\sqrt{d}}$$

In matrix notation: Q, K and V are $L \times D$

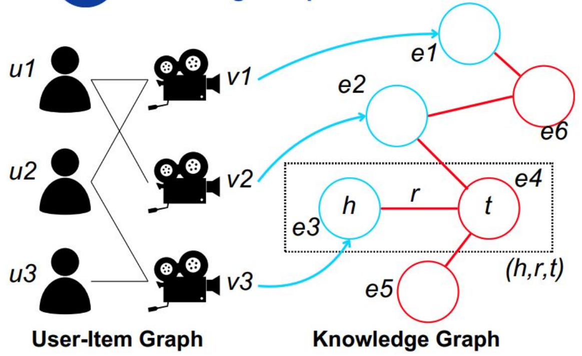
$$A(Q, K, V) = \mathrm{softmax}(QK^T)V$$

$$Q = XW_Q, K = XW_K, V = XW_V$$

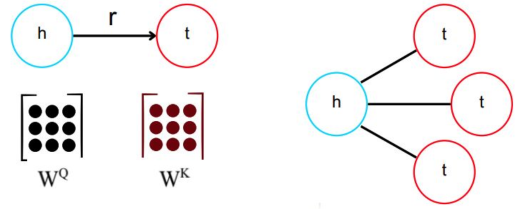
Vaswani, A. (2017). Attention is all you need. Advances in Neural Information Processing Systems.

Methodology

1 **Rationale Discovery for Knowledge Graph**



Compute rationale scores for knowledge triplets (h, r, t)

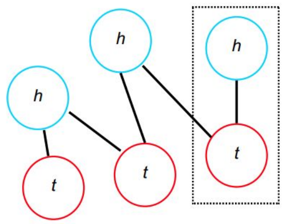


$$f(h,r,t) = \frac{e_h W_Q \cdot (e_t W_K \odot e_r)^T}{\sqrt{d}} \qquad \frac{exp f(h,r,t)}{\sum exp f(h,r,t)}$$

2 **Rationale Aware Heterogeneous Knowledge Aggregation**

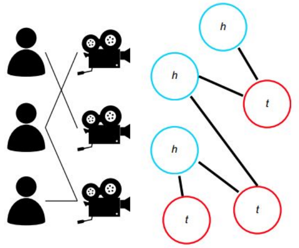
Rationale scores are incorporated into the neighbor embedding process to enhance the representation of users and entities

3 **Knowledge Aware Masked Autoencoder**



- Mask high-rationale triplets, reconstruct knowledge connections
- Train model to predict masked knowledge triplets

4 **Knowledge Rationale Aware Contrasting**



- Align KG & user-item graphs by filtering noise with rationale scores
- Train with contrastive loss on positive & negative samples

5 **Model Learning**

Joint Training & Optimization with:

- BPR Loss
- Masked Reconstruction
- Contrastive Learning

Rationale Discovery

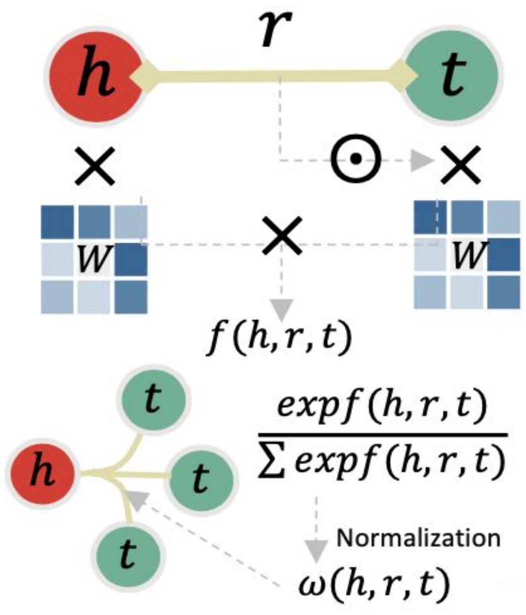
- A **rationale weighting function** assigns importance scores to knowledge triplets (head, relation, tail) using an attention mechanism inspired by heterogeneous graph transformers.

$$f(h,r,t) = \frac{e_h W_Q \cdot (e_t W_K \odot e_r)^T}{\sqrt{d}}$$

Scores are normalized across neighbors using a **softmax** function to obtain weights

- These scores help identify which triplets are most critical for understanding collaborative interactions.

$$\omega(h,r,t) = \frac{\exp\left(f(h,r,t)\right)}{\sum_{(h,r',t')\in\mathcal{N}_h} \exp\left(f\left(h,r',t'\right)\right)}$$



Rationale Discovery for Knowledge Graph

Heterogeneous Knowledge Aggregation

A knowledge aggregator embeds entities dynamically based on their importance scores, which reflect the relevance of neighboring entities in the knowledge graph. From KG $\mathcal{G}_K$,

$$\mathbf{e}_h^{(l)} = \frac{1}{|\mathcal{N}_h|} \sum_{(h,r,t)\in\mathcal{N}_h} \omega(h,r,t)\mathbf{e}_r \odot \mathbf{e}_t^{(l-1)}$$

User embeddings are derived by aggregating item embeddings from user-item interaction graphs.

From U-I Graph $\mathcal{G}_u$,

$$\mathbf{e}_u^{(l)} = \frac{1}{|\mathcal{N}_u|} \sum_{i\in\mathcal{N}_u} \mathbf{e}_v^{(l-1)}$$

$$\boxed{\mathbf{e}_h = f_k(\mathcal{G}_k; h) = \sum_l^L \mathbf{e}_h^{(l)}; \ \mathbf{e}_u = f_u(\mathcal{G}_u; u) = \sum_l^L \mathbf{e}_u^{(l)}}$$

- $e_h^{(l)}$: Embedding of entity h at layer l.
- N_h: Set of neighbors for entity h in the knowledge graph.
- $\omega(h,r,t)$: Importance score of triplet (h,r,t).
- e_r: Embedding of relation r.
- $e_t^{(l-1)}$: Embedding of entity t from the previous layer.
- e_h: Final aggregated embedding of entity h.
- $f_k(G_K; h)$: Knowledge graph aggregation function for entity h.
- $e_u^{(l)}$: Embedding of user u at layer l.
- N_u: Set of items interacted with by user u.
- $e_v^{(l-1)}$: Embedding of item v from the previous layer.
- e_u: Final aggregated embedding of user u.
- $f_u(G_U; u)$: User-item graph aggregation function for user u.

Knowledge-aware Masked Autoencoder

- **Rationale Masking Mechanism:** Masks high-importance knowledge triplets to challenge the model to reconstruct these connections.

The model selects the top-k triplets with the highest rationale scores:

$$\gamma(h,r,t) = |\mathcal{N}_h| \cdot \omega(h,r,t) = \frac{|\mathcal{N}_h| \cdot \exp\left(f(h,r,t)\right)}{\sum_{(h,r',t') \in \mathcal{N}_h} \exp\left(f\left(h,r',t'\right)\right)}$$

$$M_k = \{(h,r,t) | \gamma(h,r,t) \in \mathbf{topK}(\Gamma; k_m)\}$$

The Augmented KG: $G^k_m = G_k \setminus M_k$

- $\gamma(h,r,t)$ → Importance score of the knowledge triplet (h,r,t).
- $|N_h|$ → Number of neighboring triplets for the head entity h.
- $\omega(h,r,t)$ → Softmax-weighted score of the triplet (h,r,t).
- $f(h,r,t)$ → Scoring function that evaluates the importance of the triplet (h,r,t).
- $(h,r',t') \in N_h$ → Alternative triplets associated with head entity h (used for normalization).
- Γ → Set of **rationale scores** assigned to knowledge triplets.

Knowledge-aware Masked Autoencoder

Reconstructing with Relation-aware Objective: The model is trained to predict the missing (masked) triplets using the loss function:

$$\mathbf{e}_h = f_k(\mathcal{G}_k^m; h); \ \mathbf{e}_t = f_k(\mathcal{G}_k^m; t),$$

$f_k(.)$ = Knowledge graph aggregation function

$$\boxed{L_m = \sum_{(h,r,t)\in M_k} -\log \sigma(e_h^T \cdot (e_t \odot e_r))}$$

- M_k = Set of masked triplets.
- e_h, e_t, e_r = Learned embeddings for head, tail, and relation.
- $\sigma(\cdot)$ = Sigmoid activation function.
- $e_h^T \cdot (e_t \odot e_r)$ = Predicts whether a triplet should exist.

Knowledge Rationale-aware Contrasting

- **Augmented Graphs**: Creates noise-free views of both the knowledge graph and user-item interaction graph by removing low-importance edges.
- **Contrastive Learning**: Aligns item representations from the knowledge graph and collaborative filtering views using a contrastive objective to reinforce complementary information.
  - Uses InfoNCE loss:

$$L_c = -\sum_{v \in V} \log \frac{\text{similarity of correct pair}}{\text{similarity of correct pair} + \text{sum of incorrect pairs}}$$

$$\boxed{\mathcal{L}_c = \sum_{v \in \mathcal{V}} -\log \frac{\exp(s(\mathbf{z}_v^u, \mathbf{z}_v^k)/\tau)}{\sum_{j \in \{v, v', v''\}} (\exp(s(\mathbf{z}_v^u, \mathbf{z}_v^k)/\tau) + \exp(s(\mathbf{z}_j^u, \mathbf{z}_v^k)/\tau))}}$$

- z_v^u, z_v^k: Collaborative filtering-based and knowledge graph-based embeddings of item v.
- $s(\cdot, \cdot)$: Similarity function (cosine similarity of normalized vectors).
- τ: Temperature parameter controlling contrastive hardness.
- v', v'': Stochastically sampled negative candidates for item v.

Model Training

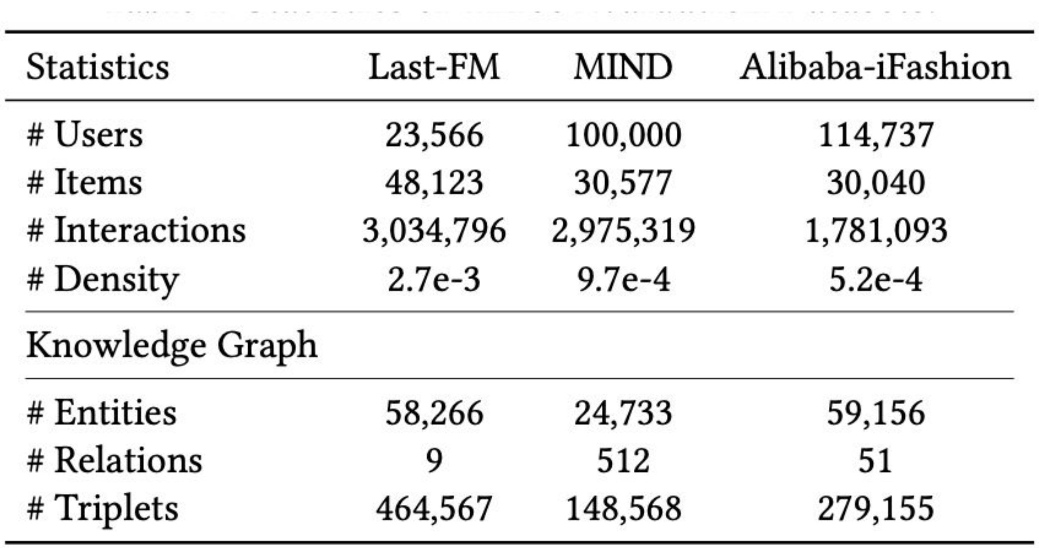
- The framework optimizes three key objectives:
  1. ζ_{rec} **Recommendation Loss** (Bayesian Personalized Ranking loss): Focused on predicting user-item interactions.

$$\mathcal{L}_{rec} = \sum_{(u,v,j)\in\mathcal{D}} -\log\sigma\left(\hat{y}_{uv} - \hat{y}_{uj}\right)$$

  2. ζ_{m} **Reconstruction Loss**: Ensures masked triplets are effectively reconstructed.
  3. ζ_{c} **Contrastive Loss**: Aligns embeddings from different graph views while minimizing noise.
- A joint loss function combines these objectives for training.

$$\mathcal{L} = \mathcal{L}_{rec} + \lambda_1\mathcal{L}_m + \lambda_2\mathcal{L}_c$$

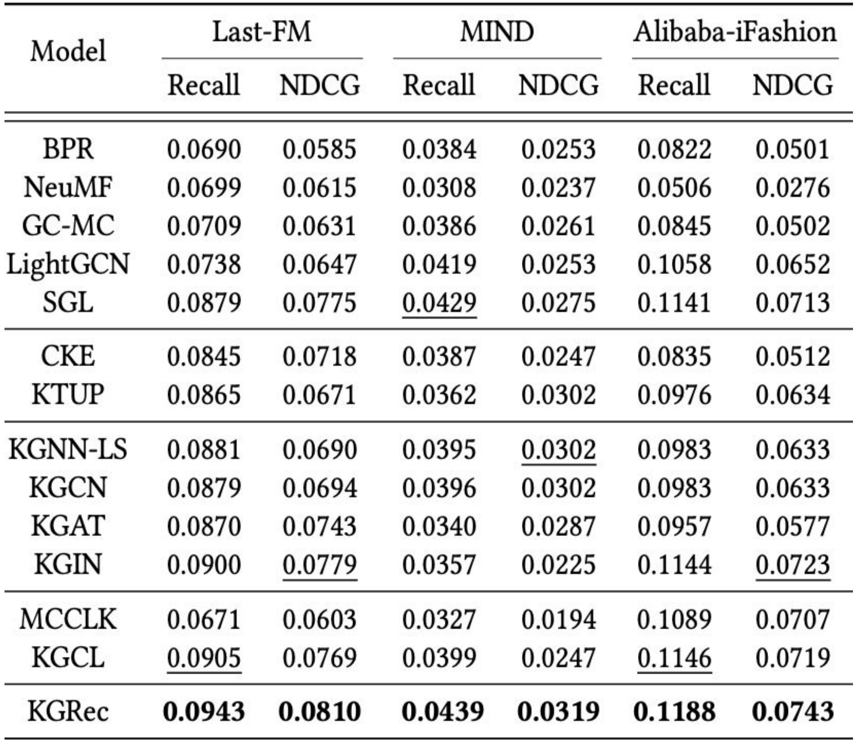
Dataset



Statistics	Last-FM	MIND	Alibaba-iFashion
# Users	23,566	100,000	114,737
# Items	48,123	30,577	30,040
# Interactions	3,034,796	2,975,319	1,781,093
# Density	2.7e-3	9.7e-4	5.2e-4
Knowledge Graph			
# Entities	58,266	24,733	59,156
# Relations	9	512	51
# Triplets	464,567	148,568	279,155

Dataset of KGRec

Experimental Result



Model	Last-FM		MIND		Alibaba-iFashion	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
BPR	0.0690	0.0585	0.0384	0.0253	0.0822	0.0501
NeuMF	0.0699	0.0615	0.0308	0.0237	0.0506	0.0276
GC-MC	0.0709	0.0631	0.0386	0.0261	0.0845	0.0502
LightGCN	0.0738	0.0647	0.0419	0.0253	0.1058	0.0652
SGL	0.0879	0.0775	0.0429	0.0275	0.1141	0.0713
CKE	0.0845	0.0718	0.0387	0.0247	0.0835	0.0512
KTUP	0.0865	0.0671	0.0362	0.0302	0.0976	0.0634
KGNN-LS	0.0881	0.0690	0.0395	0.0302	0.0983	0.0633
KGCN	0.0879	0.0694	0.0396	0.0302	0.0983	0.0633
KGAT	0.0870	0.0743	0.0340	0.0287	0.0957	0.0577
KGIN	0.0900	0.0779	0.0357	0.0225	0.1144	0.0723
MCCLK	0.0671	0.0603	0.0327	0.0194	0.1089	0.0707
KGCL	0.0905	0.0769	0.0399	0.0247	0.1146	0.0719
KGRec	**0.0943**	**0.0810**	**0.0439**	**0.0319**	**0.1188**	**0.0743**

Performance of KGRec

- KGRec outperforms all baseline models across three datasets: Last-FM, MIND, and Alibaba-iFashion.
- The **best-performing models** in each category before KGRec were:
  - **Collaborative Filtering**: LightGCN and SGL
  - **Embedding-based KG Recommenders**: CKE and KTUP
  - **GNN-based KG Recommenders**: KGIN and KGAT
  - **Self-Supervised KG Recommenders**: KGCL

Ablation Study

Ablation Setting	Last-FM (Recall)	MIND (Recall)	Alibaba-iFashion (Recall)
KGRec	**0.0943**	**0.0439**	**0.1188**
w/o MAE	0.0918	0.0374	0.1178
w/o Rationale-M	0.0929	0.0423	0.1183
w/o CL	0.0926	0.0425	0.1180

- Removing Masked Autoencoding (w/o MAE) causes the largest performance drop across all datasets, confirming that self-supervised masked knowledge reconstruction is critical.
- Random masking instead of rationale-aware masking (w/o Rationale-M) leads to lower performance, proving that identifying informative knowledge triplets improves recommendations.
- Disabling contrastive learning (w/o CL) results in performance decline, showing that cross-view knowledge-user interaction alignment is important.

Summary (KGRec)

- Leverages **self-supervised learning** to refine knowledge graph relations, enhancing interpretability and robustness.
- Introduces **rationale-aware masking, knowledge reconstruction, and contrastive learning** to improve recommendation quality.
- Outperforms state-of-the-art models on datasets like **Last-FM, MIND, and Alibaba-iFashion**, showing superior **Recall and NDCG** scores.
- Effectively handles **cold-start users and long-tail item recommendations**.

Limitations

Computational Complexity

- KGCR and KGRec require significant computational resources due to causal inference, counterfactual reasoning, and self-supervised learning.

Dependence on Knowledge Graph Quality

- The performance of both models is highly dependent on the quality and completeness of the knowledge graph. Noisy or incomplete KGs can degrade their effectiveness.

Scalability Issues

- When applied to large-scale datasets with millions of users and items, the models may face scalability challenges.

Interpretability Trade-offs

- While KGRec enhances interpretability, the rationale-aware filtering mechanism may still lack full transparency in some cases.

Future Work

Improving Scalability

- Develop more efficient algorithms and architectures to handle large-scale datasets while reducing computational overhead.

Enhancing Knowledge Graph Quality

- Investigate methods for automatically cleaning and enriching knowledge graphs to reduce noise and improve the quality of relations.

Hybrid Approaches

- Combine causal inference (KGCR) and self-supervised learning (KGRec) into a unified framework to leverage the strengths of both approaches.

Future Work

Real-World Deployment

- Implement and evaluate these models in real-world recommendation systems to assess their practical performance.

Explainability and Transparency

- Further improve interpretability by developing more transparent rationale extraction and explanation mechanisms.

Cross-Domain Recommendations

- Extend the models to support cross-domain recommendations, where knowledge from one domain can be transferred to enhance recommendations in another.

QA