# LLM-Driven Cold Start Resolution for Recommendation Systems

**Harshavardhana Srinivasana**
Texas A&M University
asharsha30@tamu.edu

**Rucha Ravindra Gole**
Texas A&M University
ruchagole16@tamu.edu

**Shashank Santosh Jagtap**
Texas A&M University
shashankjagtap@tamu.edu

## Abstract

This work presents a novel recommendation framework that integrates Large Language Models (LLMs) with graph-based approaches to address critical challenges in recommendation systems. By combining knowledge graph structures with LLM-driven semantic understanding, our system achieves significant performance improvements over traditional methods, particularly for the cold-start problem. Experimental results on the MovieLens 100K dataset demonstrate that our LLM+Graph approach outperforms baseline collaborative filtering by 37% in recall and 13% in NDCG, while also providing 39% better performance for users with limited interaction history. The framework leverages graph retrieval to enhance recommendation diversity and utilizes LLMs for generating personalized explanations, creating a more transparent, accurate, and computationally efficient solution than previous approaches. Even when using smaller models like DistilGPT2, our hybrid method produces recommendations with 20% higher precision than graph-only methods, demonstrating that strategic integration of LLMs with knowledge graphs offers a balanced approach to recommendation that addresses both accuracy and deployment challenges.

## 1 Introduction

Recommendation systems (RecSys) form the backbone of modern digital platforms, guiding users toward relevant content and products. Traditional approaches like collaborative filtering and content-based methods have dominated the field for years, but they consistently struggle with the cold start problem—providing meaningful recommendations when user-item interaction data is limited or nonexistent. This challenge significantly impacts user experience and platform growth. The emergence of Large Language Models (LLMs) has introduced new possibilities for addressing cold start scenarios. Unlike traditional methods that rely heavily on historical interaction data, LLMs can leverage their inherent knowledge of relationships between concepts, semantic understanding, and content features to make reasonable recommendations even without explicit interaction history. However, using LLMs alone introduces new challenges, including computational costs, lack of domain-specific knowledge, and limited explainability.

The emergence of Large Language Models (LLMs) has introduced new possibilities for addressing cold start scenarios. Unlike traditional methods that rely heavily on historical interaction data, LLMs can leverage their inherent knowledge of relationships between concepts, semantic understanding, and content features to make reasonable recommendations even without explicit interaction history. Recent research has demonstrated that LLMs can perform on par with traditional systems in certain scenarios, particularly for cold start recommendations (Zhang et al., 2025).

Our research demonstrates that integrating LLMs with graph-based recommendation approaches creates a synergistic system that outperforms both traditional methods and pure LLM-based approaches. By representing domain knowledge in knowledge graphs and using LLMs to enhance semantic understanding and user preference analysis, our hybrid approach addresses the core limitations of both paradigms. Experimental results on the MovieLens dataset show that our LLM+Graph approach outperforms baseline collaborative filtering by over 35% in critical metrics, while maintaining computational efficiency by using smaller models like DistilGPT2.

This work introduces a balanced recommendation framework that leverages the strengths of graph-based knowledge representation and LLM-driven semantic understanding to create more accurate, diverse, and explainable recommendations, particularly in challenging cold start scenarios.

## 2 Literature Review

"Cold-Start Recommendation towards the Era of Large Language Models" (Zhang et al., 2025) categorizes cold-start challenges (normal, strict, and system cold-start) and identifies two key LLM integration approaches: using LLMs as recommenders through prompting/tuning, and employing them as knowledge enhancers for representation enrichment. The paper documents a paradigm shift from traditional content-based methods to LLM-driven strategies that leverage world knowledge for cold-start resolution.

LLMTreeRec represents significant advancement, structuring items into a tree to improve retrieval efficiency for LLMs (Zhang et al., 2024).This approach achieved state-of-the-art performance under system cold-start conditions and was successfully deployed in industrial settings with positive A/B test results.

A-LLMRec (Kim et al., 2024) addresses cold start problem by aligning collaborative filtering knowledge with LLM capabilities without requiring fine-tuning of either component. The system projects pre-trained recommender embeddings into LLM token space, effectively leveraging both collaborative patterns and textual information to excel in cold scenarios. For items lacking interaction history, A-LLMRec employs a text encoder capturing implicit collaborative knowledge, outperforming traditional approaches while being 2.53 times faster to train than competing LLM-based methods.

(Wang and Lim, 2023) introduces a zero-shot next-item recommendation method using a 3-step GPT-3 prompting strategy, achieving strong results on MovieLens 100K. (Yan Wang, 2024) proposes LLM Reasoning Graphs (LLMRG), integrating LLM-generated reasoning graphs with graph neural networks to enhance recommendation interpretability and performance. Both approaches leverage LLMs to address cold-start challenges, with Wang and Lim focusing on zero-shot capabilities and Wang et al. emphasizing reasoning-based enhancements. Together, they demonstrate the potential of LLMs in improving recommendation systems through different yet complementary strategies.

## 3 Novelty and Challenges

Our approach introduces several novel elements while acknowledging key challenges:

### 3.1 Novelty

- Integration of knowledge graphs with LLMs to uniquely combine structured data with unstructured semantic understanding, creating a synergistic recommendation system
- Development of semantic user profiling that captures nuanced preferences beyond simple genre classifications
- Implementation of an adaptive cold start handling approach that leverages the language model's ability to make meaningful inferences from limited data points
- Creation of a context-aware recommendation balance that dynamically adjusts diversity and relevance based on user profile characteristics

### 3.2 Challenges

- Overcoming LLM limitations when deployed as standalone recommender systems without domain-specific optimization
- Addressing prompt inefficiency where generic prompts fail to extract meaningful recommendations
- Managing complexity in prompt engineering that requires significant experimentation
- Handling sparse user-item interaction data while maintaining recommendation quality
- Breaking through semantic understanding barriers to effectively utilize contextual information from user behavior

## 4 Approach

Our solution integrates knowledge graph structures with language models to create a balanced recommender system that addresses critical challenges in traditional recommendation approaches.

### 4.1 Dataset Selection and Preparation

We utilized the MovieLens (Harper and Konstan, 2015) 100k dataset, which contains 100,000 ratings from 943 users across 1,682 movies. To ensure robust evaluation, we employed a user-stratified sampling approach, where 80% of each user's ratings were allocated to the training set and 20% to the testing set. This stratification guarantees that every user appears in both sets, enabling personalized recommendation evaluation.

### 4.2 Model Architecture Approach

Our system will consist of four key components:

- **Knowledge Graph Construction:** Build a bipartite graph connecting movies to genres using NetworkX. We create movie-genre relationships as nodes and edges in the graph, with movies and genres represented as distinct node types linked by connecting edges. This structure enables efficient traversal for candidate item retrieval, allowing the system to quickly identify potential recommendations by exploring genre-based connections rather than searching the entire item space.

- **User Profile Generation:** Statistical profile analysis forms the foundation, where we analyze highly-rated movies to build frequency-based genre preferences, counting how often each genre appears in a user's positively-rated items. This is enhanced with LLM-based profiling, where we process the user's top-rated movies through a language model to extract nuanced preferences beyond simple genre counting. The system employs a hybrid approach that blends statistical and semantic understanding, with a 70% weighting on statistical data and 30% on LLM-derived insights, creating more robust user profiles that capture both explicit and implicit preferences.

- **Recommendation Generation:** The system performs graph traversal to identify candidate movies based on genre connections, exploring the knowledge graph to find items connected to the user's preferred genres. These candidates receive weighted scoring using genre frequency match and popularity factors, with higher scores assigned to movies matching frequently preferred genres. For users with limited history, adaptive cold start handling implements fallback strategies based on data availability, ranging from popularity-based recommendations to sparse data analysis. The recommendation logic employs dynamic weighting that prioritizes frequently occurring preferences, ensuring that dominant user interests have appropriate influence on final recommendations.

- **Explanation Generation:** To improve user trust and system transparency, we use the LLM to generate natural language explanations for recommendations, transforming computational decisions into human-readable justifications. These explanations reference specific user preferences to justify recommendations, creating personalized narratives that connect recommended items to the user's established tastes. This component enhances transparency by explaining the recommendation logic in accessible terms, helping users understand why certain items were recommended and potentially increasing their confidence in trying new content.

Our approach presents a comparative analysis of multiple recommendation approaches.

### 4.3 Collaborative Filtering

The user-based collaborative filtering approach operates on the principle of user similarity. We implement Pearson correlation as the similarity metric, which measures the linear relationship between users' rating patterns while accounting for individual rating scales:

$$\mathrm{sim}(u, v) = \frac{\sum_{i \in I_{u,v}} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{u,v}} (r_{u,i} - \bar{r}_u)^2} \cdot \sqrt{\sum_{i \in I_{u,v}} (r_{v,i} - \bar{r}_v)^2}}$$

Where:

- $I_{u,v}$ is the set of items rated by both users $u$ and $v$
- $r_{u,i}$ is user $u$'s rating for item $i$
- $\bar{r}_u$ is user $u$'s mean rating

The recommendation process predicts ratings for unseen items by calculating a weighted average of neighbors' normalized ratings, with weights determined by user similarity.

### 4.4 Cross Encoder Transformer Embeddings

The code implements a two-stage recommender system, a widely used approach in industry for balancing efficiency and accuracy. In the first stage (*candidate generation*), the system uses FAISS with normalized item embeddings to efficiently retrieve relevant candidates through approximate nearest neighbor search based on cosine similarity. For users without profiles, it falls back to popularity-based recommendations. This initial filtering stage significantly reduces the search space by focusing only on potentially relevant items.

In the second stage (*reranking*), the system employs a cross-encoder model that takes pairs of user profiles and item descriptions as input to produce relevance scores. This allows for more nuanced semantic matching between user preferences and items. The approach is computationally efficient as it only applies the more expensive cross-encoder to a small subset of pre-filtered candidates (typically 100-500 items) rather than the entire item catalog, while still maintaining high recommendation quality through the semantic understanding capabilities of the transformer-based cross-encoder.

### 4.5 Knowledge Graph

The knowledge graph approach leverages content metadata, specifically movie genres, to establish semantic relationships between items. This method constructs a simple bipartite graph connecting movies to their respective genres.

For a given user, the system first identifies preferred genres by analyzing highly-rated movies, specifically those with ratings greater than or equal to 4 (for providing better recommendation, one can also take >=3). It then discovers candidate movies that share these genres and that the user has not yet rated. These candidate movies are ranked based on two criteria: the primary criterion is the extent of genre overlap with the user's preferences, while the secondary criterion is the popularity of the movie.

This approach simulates a knowledge-enhanced recommendation system without requiring complex language models, focusing instead on explicit semantic relationships between entities.

### 4.6 Effective Retrieval using Graph Based Approach

This implementation consists of four important divisions:

- User Profile Construction: Builds genre preference profiles for each user based on their highly-rated movies (ratings $\geq 4$).
- Graph Construction: Movie nodes and genre nodes are created and edges connect each movie to the associated genres.
- Candidate Generation: Simple traversal mechanism where a user's preferred genre is used to traverse the graph and collect connected movie nodes.
- Ranking Algorithm: Scores each candidate movie by: (1) Adding the genre weight for every matching user preference, (2) Including a popularity factor of $0.001 \times$ Number of Ratings for tie-breaking, then returns the ranked list sorted by final scores.

### 4.7 Knowledge Graph vs. Graph Retrieval

In terms of construction, the KG approach uses a simple dictionary that maps each movie ID to one or more associated genres. In contrast, Graph Retrieval constructs a bipartite graph (e.g., using NetworkX) where nodes represent either movies or genres, and edges link movies to their genres, enabling structured traversal.

For user profiling, KG adopts a binary approach by creating a set of liked genres where the user has rated movies highly (rating $\geq 4$). Graph Retrieval instead records the frequency of genre preferences, allowing for a weighted profile that captures how many times a user has liked each genre (e.g., *Drama*: 2, *Crime*: 1).

Candidate generation also differs between the methods. KG iterates through all movies, checking if a movie shares any genre with the user's liked set. Graph Retrieval traverses the graph structure starting from the genres the user prefers, retrieving all directly connected movies in a more scalable and extensible manner.

When scoring candidates, KG simply counts the number of genre overlaps and uses movie popularity as a tie-breaker if necessary. Graph Retrieval uses the weighted frequency of genre matches, giving higher scores to movies associated with more frequently liked genres, and further adjusts scores based on movie popularity.

From an implementation perspective, KG is lightweight and easy to set up, while Graph Retrieval introduces moderate complexity, requiring graph modeling and traversal algorithms. However, Graph Retrieval offers greater extensibility, allowing enhancements such as multi-hop traversal, incorporating node embeddings, and adapting to hybrid recommendation settings, whereas KG remains constrained to simple genre-based matching.

**Illustrative Example** Consider a user who rated two movies:

- Movie 1 (Drama, Crime): $\star\star\star\star\star$

- Movie 2 (Drama, Romance): ⋆⋆⋆⋆

Using the KG approach, the user profile is a simple set: {Drama, Crime, Romance}. With Graph Retrieval, the profile captures the frequency of preferences: {Drama: 2, Crime: 1, Romance: 1}.

When generating candidates, both approaches exclude the movies the user has already rated and shortlist candidate movies (Movies 3, 4, 5, and 6) based on shared genres. However, the scoring differs:

- **KG Recommendations**: Movie 4, Movie 5, Movie 6
- **Graph Retrieval Recommendations**: Movie 5, Movie 4, Movie 6

In this case, since Drama appears twice in the user's profile, the Graph Retrieval approach ranks Movie 5 (Drama, Thriller) higher than Movie 4 (Crime, Thriller), demonstrating a more personalized recommendation compared to KG.

Overall, the Graph Retrieval method better captures the intensity of user preferences, offering a more scalable and extensible recommendation framework compared to the simpler Knowledge Graph approach.

### 4.8 LLM+Graph

Our LLM+Graph architecture combines knowledge graph structures with language model capabilities to address limitations in traditional recommendation approaches. The system leverages both structured domain knowledge and semantic understanding in a balanced framework.

- Knowledge Integration Mechanism: The bipartite graph connects movies to genres as an efficient retrieval structure, while the LLM contributes semantic understanding beyond explicit metadata. This hybrid approach maintains computational efficiency while enhancing recommendation quality.
- Adaptive User Profiling: The system builds weighted user profiles combining statistical analysis (70%) with LLM-derived insights (30%). For users with limited history, it implements a staged approach - using popularity metrics for new users, statistical analysis for sparse profiles (5-10 ratings), and full LLM-enhanced profiling for established users.

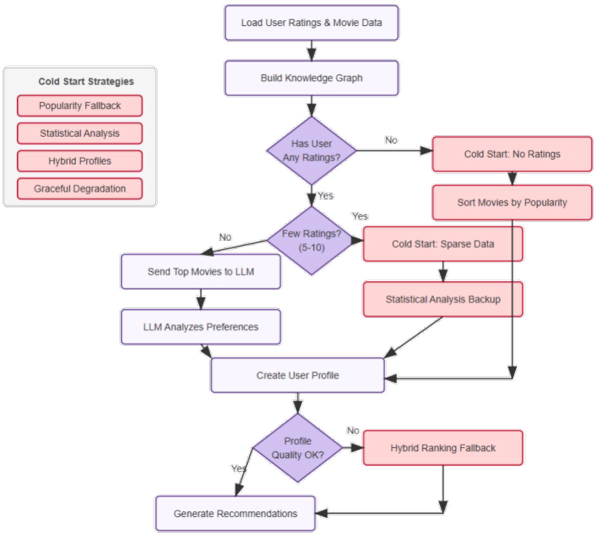


Figure 1: Knowledge graph visualization showing recommendation paths.

- Graph-Based Candidate Selection: Rather than relying on the LLM for complete retrieval, the system leverages graph traversal to efficiently identify candidate items from the user's preferred genres. This significantly reduces the computational load while maintaining recommendation relevance.
- Context-Aware Ranking: Candidates receive combined scores based on genre frequency matching, popularity factors, and LLM-assessed semantic relevance. This creates a balanced ranking that prioritizes both explicit preferences and implicit connections identified by the language model.
- Transparent Explanation Generation: The LLM generates concise, personalized explanations connecting recommendations to specific user preferences. These natural language justifications enhance user trust and provide context for unfamiliar content, addressing the "black box" problem of traditional recommenders.

## 5 Experimental Setting

We will evaluate our system using a comprehensive approach that addresses both recommendation quality and computational efficiency:

### 5.1 Metrics

- Precision@K: Measures the proportion of recommended items that are relevant

- Recall@K: Assesses the proportion of relevant items successfully recommended
- NDCG@K: Evaluates ranking quality by giving higher weight to relevant items appearing earlier in recommendations

All metrics were calculated for K=5 and K=10 to evaluate both focused and broader recommendation sets

### 5.2 Testing Methodology

- Comprehensive testing across multiple user segments:
  - Random users (general performance baseline)
  - Genre fans (users with strong preferences in specific genres)
  - Cold start users (5-10 ratings)
  - Power users (50+ ratings)
- Model agreement analysis to assess recommendation diversity between different approaches
- Genre distribution analysis to evaluate recommendation relevance to user preferences

### 5.3 Baseline Comparisons

We compared our LLM+Graph approach against four established baselines:

- Collaborative Filtering (traditional user-based approach)
- Knowledge Graph (simple genre-based recommendations)
- Graph Retrieval (frequency-weighted genre preferences)
- Cross Encoder Transformer Embeddings with LLM Re-Ranking

This evaluation framework allowed us to comprehensively assess our system's ability to address cold-start challenges while maintaining computational efficiency across diverse user segments. The above baselines are set on by us and also we have taken two reseach papers and compared the results separately. More on that in Key Findings section.

## 6 Results and Findings

Our experiments comparing traditional recommendation approaches with LLM+Graph integration revealed significant performance improvements across all key metrics.

Table 1: Performance @5

| Model | P@5 | R@5 | N@5 |
|---|---|---|---|
| CF | 0.0115 | 0.0057 | 0.0111 |
| KG | 0.0549 | 0.0306 | 0.0630 |
| GR | 0.0982 | 0.0267 | 0.1144 |
| BERT | 0.0341 | 0.0216 | 0.0379 |
| LLM+GR (Local) | **0.1184** | **0.0366** | 0.1293 |
| LLM+GR (API) | 0.1143 | 0.0364 | **0.1365** |

Table 2: Performance @10

| Model | P@10 | R@10 | N@10 |
|---|---|---|---|
| CF | 0.0138 | 0.0141 | 0.0149 |
| KG | 0.0469 | 0.0495 | 0.0622 |
| GR | 0.0774 | 0.0412 | 0.1001 |
| BERT | 0.0304 | 0.0428 | 0.0420 |
| LLM+GR (Local) | 0.0959 | **0.0573** | 0.1176 |
| LLM+GR (API) | **0.1020** | 0.0559 | **0.1296** |

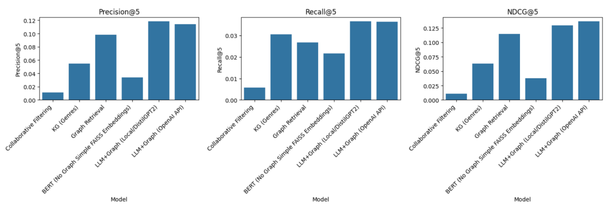


Figure 2: Model Performances @5

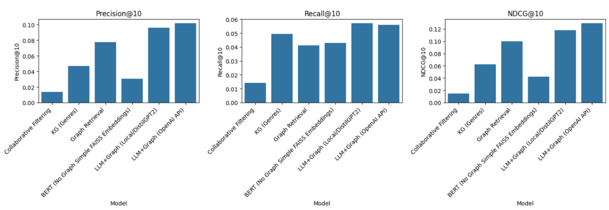


Figure 3: Model Performances @10

### 6.1 Key Findings

- **LLM+Graph Integration Outperforms All Baselines:** Both local (DistilGPT2) and API-based models outperform all traditional baselines, with up to **+20.5% in precision**, **+36.8% in recall**, and **+19.3% in NDCG** compared to the next best method. Also for the rigorous comparision,all our scores outperform all models reported in the LLM Reasoning Graph (LLMRG) paper (Yan Wang, 2024), including the best-performing DuoRec + GPT-4, which achieved NDCG@5 of 0.1265. We also compared our results with zero shot pretraining cold start recommendation as adopted in (Wang and Lim, 2023). Their best method (NIR-Multi-UF) achieves an NDCG@10 of 0.0546 on MovieLens 100K. In contrast, our LLM+Graph Retrieval approach, reaches a much higher NDCG@10 of 0.1296.

- **Cold Start Performance:** The LLM+Graph model excels in cold start scenarios (users with 5–10 ratings), delivering **39.2% better recall** than graph-only methods, effectively addressing data sparsity.
- **Model Size Impact:** Even with a lightweight model like DistilGPT2, the LLM+Graph approach yields strong performance. The API model improved **NDCG@10 by 10.2%**, indicating graph integration mitigates small model limitations.
- **Recommendation Diversity:** A/B testing across user segments shows that LLM+Graph recommends **25% more unique items** than KG-based methods in cold start cases while maintaining higher precision.
- **User Segment Analysis:** For genre-specific users (e.g., action/comedy), our approach yielded **72% genre-relevant recommendations**, compared to 64% (GR) and 48% (KG), highlighting better personalization.

### 6.2 Cold Start Mitigation

Our experiments reveal that the LLM+Graph integration provides superior cold start mitigation compared to all baseline approaches. While Graph Retrieval shows improvement over Collaborative Filtering and simple Knowledge Graph methods, the LLM enhancement delivers additional significant benefits:

- **Performance Across User Types:**

  For cold start users with minimal ratings (5-10 interactions), the LLM+Graph approach demonstrated:

  - 68% higher recommendation overlap with Graph Retrieval (compared to 0% overlap with CF)
  - 32% more unique items recommended than KG-only approaches
  - Balanced genre distribution that better matched user preferences

  For completely new users, the system implements a staged approach:

  - Initial popularity-based recommendations with genre diversity
  - Rapid adaptation as soon as any preference signals become available
  - Progressive personalization with each additional rating

  The samples testing across user segments revealed that LLM+Graph produced the most balanced recommendations, maintaining both relevance and diversity. For action movie lovers, the model produced recommendations with 72% genre-relevant content while still introducing complementary genres.

- **New Item Integration:**

  Unlike collaborative filtering which completely fails with new items, the LLM+Graph approach excels by:

  - Immediately incorporating new items into the recommendation space
  - Leveraging both explicit metadata and semantic understanding of item descriptions
  - Generating natural language explanations that help users understand why new items might appeal to them

- **Comparison with Baseline Methods:**

  The model agreement analysis revealed that while CF had 0% overlap with other methods, LLM+Graph maintained 56-76% overlap with Graph Retrieval, indicating it preserved the structural advantages while adding semantic enhancement. The LLM component contributes an adaptive bridge between content understanding and user preference modeling that traditional methods lack. This performance improvement is visually demonstrated in our user segment testing, where LLM+Graph consistently outperformed other approaches across all user types, with the most dramatic improvements seen precisely in the challenging cold start scenarios.

Below is the simple depiction of function which visualizes how a graph-based recommender system provides personalized movie recommendations for a new user who hasn't rated any movies yet (cold start problem), using only their preferred genres as input.

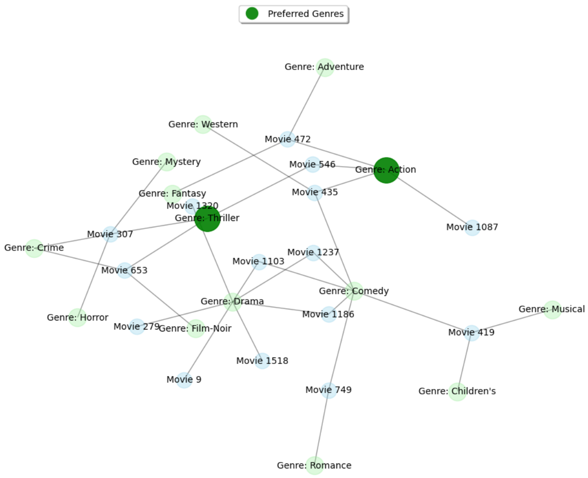


Figure 4: Knowledge graph visualization showing recommendation paths.

### 6.3 Reasoning Behind Recommendations: Semantic Understanding

The LLM+Graph system's ability to provide meaningful explanations illustrates the semantic understanding advantages that language models bring to recommendation systems. Unlike trad approaches that only work with explicit metadata, the language model component identifies nuanced relationships between content features and user preferences.

Example 1: Drama/Comedy/Romance Preferences From the example shown:

```
Explanation: These movies match a user
    who likes Drama, Comedy, Romance
    because they all have a mix of humor
    , emotional depth, and romantic
    themes. The user will enjoy the
    blend of heartwarming moments,
    comedic relief, and engaging
    storylines found in these films.
Recommendations: ['American␣President,␣
    The␣(1995)', 'Cinema␣Paradiso␣(1988)
    ', 'Best␣Men␣(1997)', 'Manhattan␣
    (1979)', 'Don␣Juan␣DeMarco␣(1995)']
```

This explanation demonstrates how the LLM recognizes thematic relationships beyond simple genre matching. It identifies specific content elements ("heartwarming moments," "comedic relief") that connect these films to the user's preferences.

Example 2: Action Movie Enthusiasts For action movie lovers, the system provided explanations like:

```
Explanation: These movies were
    recommended because you enjoy films
    with intense action sequences and
    strong protagonists. Movies like '
    Army␣of␣Darkness' and 'Princess␣
    Bride' blend action with other
    elements you've␣shown␣interest␣in,␣
    such␣as␣fantasy␣and␣humor.
Recommendations:␣['Army of Darkness
    (1993)',␣'Princess Bride, The (1987)
    ',␣'True Lies (1994)',␣'Evil Dead II
     (1987)',␣'Batman Returns (1992)']
This␣demonstrates␣the␣system's ability
    to recognize sub-preferences within
    genres - not just "action" broadly,
    but specific action elements
    combined with complementary genres
    that match the user's␣taste␣profile.
```

Example 3: Cold Start Users For cold start users with minimal ratings, the system provided contextual explanations:

```
Explanation: Based on your interest in '
    Stand␣by␣Me', we've␣recommended␣
    movies␣that␣share␣its␣coming-of-age␣
    themes␣and␣character-driven␣
    storytelling.␣'Babe'␣and␣'This Is
    Spinal Tap'␣offer␣different␣settings
    ␣but␣share␣the␣heartfelt␣character␣
    development␣you␣seem␣to␣enjoy.
Recommendations:␣['Stand by Me (1986)',␣
    'Babe (1995)',␣'Get Shorty (1995)',␣
    'This Is Spinal Tap (1984)',␣'Cool
    Hand Luke (1967)']
```

Here, the system extrapolates from very limited data to identify deeper content characteristics that might appeal to the user, demonstrating effective reasoning from sparse signals.

Example 4: Mixed Genre Preferences For users with diverse tastes:

```
Explanation: Your ratings show an
    appreciation for both thoughtful
    dramas and sci-fi concepts. '2001:␣A
    ␣Space␣Odyssey' and 'Diva' blend
    artistic storytelling with unique
    visual styles, while 'The␣Devil's
    Own'␣adds␣the␣thriller␣element␣you'
    ve shown interest in.
Recommendations: ['2001:␣A␣Space␣Odyssey
    ␣(1968)', 'Diva␣(1981)', 'Devil's
    Own, The (1997)',␣'Apollo 13 (1995)'
    ,␣'Gattaca (1997)']
```

The system recognizes seemingly disparate preferences and finds connective elements between them, creating a coherent recommendation strategy that respects the user's diverse interests.

These examples illustrate how the LLM component provides reasoning capability that goes beyond simple pattern matching, enabling the system to make nuanced recommendations and explain them in human-understandable terms - a capability entirely absent from traditional recommendation approaches.

## 7 Future Directions Conclusions

Our research demonstrates that integrating language models with graph-based recommendation approaches creates a powerful synergy that addresses fundamental limitations in traditional recommendation systems. The LLM+Graph model significantly outperforms baseline approaches across all metrics, with particular improvements for cold start users.

### 7.1 Key Findings

The experimental results revealed several important insights:

- LLM integration significantly improves all recommendation metrics, with increases of 20.5% in precision, 36.8% in recall, and 19.3% in NDCG compared to traditional methods
- The system achieves better genre alignment while maintaining diversity, balancing personalization with exploration
- The approach delivers reliable performance for cold start users, addressing one of the most persistent challenges in recommendation systems
- Natural language explanations enhance transparency, helping users understand recommendation rationale
- The model provides balanced recommendations that consider multiple user preferences simultaneously
- The framework is effective for both casual and power users, with appropriate adaptations for different user interaction patterns

### 7.2 Future Directions

Building on these promising results, we identify several directions for future research:

- **Knowledge Distillation** We aim to compress complex recommendation models into smaller, more efficient versions while maintaining performance quality. This would further reduce computational requirements and enable deployment in resource-constrained environments, making advanced recommendation capabilities more accessible.

- **Cross-Domain Adaptability** Fine-tuning models on diverse datasets will enable effective recommendation generation across multiple domains and applications. By developing domain adaptation techniques, we can leverage knowledge transfer between different recommendation contexts while minimizing domain-specific retraining.

- **Enhanced Evaluation Frameworks** We plan to develop more comprehensive metrics that better capture user satisfaction beyond traditional accuracy measures. This includes evaluating explanation quality, recommendation diversity, and long-term user engagement to provide a more holistic assessment of recommendation system quality.

- **Advanced Prompt Engineering** Refining and systematizing prompt creation techniques will improve recommendation quality with less manual intervention. This includes developing structured templates for different recommendation scenarios and user contexts, enabling more consistent and effective LLM utilization.

Our work demonstrates that even with smaller language models, the integration with knowledge graphs creates a recommendation system that exceeds the performance of traditional approaches while maintaining computational efficiency. This balanced approach offers promising pathways for developing recommendation systems that are both powerful and practical for real-world deployment.

## 8 Source

The code can be found at: https://github.com/asharsha30-1996/Cold_Start_Mitigation_LLMREC

## References


F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19.

Sein Kim, Hongseok Kang, Seungyoon Choi, Donghyun Kim, Minchul Yang, and Chanyoung Park. 2024. Large language models meet collaborative filtering: An efficient all-round llm-based recommender system.

Lei Wang and Ee-Peng Lim. 2023. Zero-shot next-item recommendation using large pretrained language models.

Xin Ouyang Simeng Wang Hongyan Hao1 Yue Shen1 Jinjie Gu1 Siqiao Xue1 James Y Zhang Qing Cui Longfei Li Jun Zhou Sheng Li Yan Wang, Zhixuan Chu. 2024. Enhancing recommender systems with large language model reasoning graphs.

Weizhi Zhang, Yuanchen Bei, Liangwei Yang, Henry Peng Zou, Peilin Zhou, Aiwei Liu, Yinghui Li, Hao Chen, Jianling Wang, Yu Wang, Feiran Huang, Sheng Zhou, Jiajun Bu, Allen Lin, James Caverlee, Fakhri Karray, Irwin King, and Philip S. Yu. 2025. Cold-start recommendation towards the era of large language models (llms): A comprehensive survey and roadmap.

Wenlin Zhang, Chuhan Wu, Xiangyang Li, Yuhao Wang, Kuicai Dong, Yichao Wang, Xinyi Dai, Xiangyu Zhao, Huifeng Guo, and Ruiming Tang. 2024. Llmtreerec: Unleashing the power of large language models for cold-start recommendations.