GenRecGraph

Generative Graph Models for Cold-Start in Recommendation Systems

GNN+XAI M2025

Dr. Amilpur Santhosh

ShreeRaj Mummidivarapu
S20220010144

Rahul Tarachand
S20220010223

Presentation Outline

1. Problem Definition & Motivation
2. Literature Review
3. Dataset Selection Preprocessing
4. Model Architecture & Implementation
5. Results, Evaluation Metrics & Analysis
6. Innovation & Original Contribution
7. Conclusion

Clarity of the Problem Statement

1. The Core Issue: Cold-Start

- **Dependency:** Recommender systems usually require user history to function.
- **The Gap:** New users have **zero history**.
- **Failure:** Traditional AI (like Matrix Factorization) cannot make predictions for these "invisible" users.

2. Our Research Objective

- Build a **Generative Graph Model** that accepts raw user profiles.
- **Generate likely connections** instantly, bridging the gap before the user interacts.
- Ensure the system works effectively even on sparse datasets.

Motivation for Choosing the Problem

Why This Matters

- **Real-World Pain:** Platforms like Netflix and Amazon lose new users if the first recommendations are irrelevant.
- **Current Limitation:** Standard methods struggle to generalize for completely new entities.

Why Graph Generative Models?

- **Rich Context:** Graphs capture complex relationships between users and items better than simple tables.
- **Smart Prediction:** Generative models learn the *probability* of a connection, not just a yes/no match.

Expected Impact:
A smoother experience for new users and improved recommendations for new users.

Literature Review & Related Work

1. **LightGCN: Simplifying Graph Convolution (He et al.)**
   - **Concept:** A streamlined model that removes unnecessary complexity (like non-linearities) to train faster.
   - **Relevance:** It is the current industry standard for efficiency, though it typically requires adaptation to handle new users.
2. **Web-Scale Recommender Systems (PinSage - Ying et al.)**
   - **Concept:** A highly scalable system built for Pinterest that uses random walks to process billions of items.
   - **Relevance:** Demonstrated that Graph Neural Networks can work effectively in massive, real-world commercial applications.
3. **Inductive Representation Learning (GraphSAGE - Hamilton et al.)**
   - **Concept:** Introduced "Inductive Learning," allowing the model to generate embeddings for nodes it has never seen before.
   - **Relevance:** This architecture forms the foundational backbone of our project, enabling us to solve the Cold-Start problem.

Dataset Selection & Feature Engineering

1. Dataset: MovieLens-25M

- **Source:** GroupLens Research (25 million ratings).
- **Why:** Standard benchmark with rich metadata (Genres, Timestamps) crucial for graph learning.
- **Filtering:** We remove noise by discarding ratings < 3.0 and pruning inactive users (< 20 interactions).

2. Feature Engineering (The Input Signals)

- **User Features:** We construct a 5-dimensional statistical vector (Mean rating, Variance, Count) to represent behavior.
- **Movie Features:** We use Multi-hot encoding for **Genres** and normalized values for **Release Year**.
- **Significance:** These features allow the model to understand "content" even before it sees any graph connections.

Graph Construction & Splitting Strategy

3. Graph Construction

- **Structure:** A **Bipartite Graph** connecting Users $\leftrightarrow$ Movies.
- **Edges:** represent positive interactions (Ratings ≥ 3).
- **Leakage Prevention:** Only *Training* edges exist in the graph structure. Validation/Test edges are hidden during message passing.

4. Temporal Splitting Strategy

- **Time-Based Split:** We sort interactions by timestamp (70% Train / 10% Val / 20% Test).
- **Why:** Random splitting "cheats" by using future data to predict the past. Temporal splitting simulates real-world forecasting.
- **Cold-Start Isolation:** Users appearing *only* in the Test set are flagged to specifically evaluate Zero-Shot performance.

1. Core Architecture: Graph Autoencoder (GAE)

The Modular Framework

- We implemented a modular **Encoder-Decoder** design pattern.
- **Goal:** To compress users/movies into "Embeddings" (Encoder) and use them to predict missing ratings (Decoder).

The Pipeline Flow

1. **Input:** Graph Structure (A) + Node Features (X).
2. **Encode:** $Z = Encoder(X, A)$. Maps nodes to a low-dimensional latent space.
3. **Decode:** $\hat{A} = Decoder(Z)$. Reconstructs the adjacency matrix (predicts the rating).
4. **Loss:** We optimize using Binary Cross Entropy (BCE) to minimize prediction error.

The Encoders (Learning Representations)

The Selected Model: GraphSAGE (Inductive)

- **Mechanism:** Instead of memorizing user IDs, it learns a function to **aggregate** features from local neighbors.
- **Why it Won:** It supports "Inductive Learning," meaning it can generate embeddings for entirely new users instantly, which is essential for solving Cold-Start.

Comparative Baselines

- **GCN (Graph Convolutional Network):** The standard baseline. It is "Transductive," meaning it requires a fixed graph structure and cannot handle new nodes without retraining.
- **GAT (Graph Attention Network):** Uses an attention mechanism to weigh the importance of different neighbors. It offers high accuracy but is computationally expensive.

The Decoders (Predicting Interactions)

The Selected Model: MLP Decoder

- **Mechanism:** Concatenates User and Movie vectors and passes them through a deep neural network (Dense Layers $\rightarrow$ ReLU).
- **Advantage:** It captures complex, non-linear relationships between users and genres that simple math cannot find.

Comparative Baselines

- **Bilinear Decoder:** Learns a weighted interaction matrix. It is more expressive than a dot product but less flexible than an MLP.
- **GraphVAE (Variational Autoencoder):** A generative model that learns a probabilistic distribution. Excellent for generating graph structures but slightly less precise for ranking.

Cold-Start Implementation & Inference

Handling Zero-Shot Cold-Start

- **The Challenge:** A brand new user has no edges, so GraphSAGE has no neighbors to aggregate.
- **The Solution (Feature Projection):** We implemented a specialized linear layer that projects raw profile data (Age, Gender) directly into the embedding space.
- **Hybrid Blending:** The system calculates a weighted average of this projection and similar users to create a stable initial embedding.

The Recommendation Engine

- **Diversity Logic:** We applied **Maximal Marginal Relevance (MMR)** in the final ranking.
- **Impact:** This ensures the top 10 recommendations aren't just 10 movies from the same series, but a diverse mix of the user's interests.

1. Evaluation Metrics

How We Measure Success

AUC (Area Under ROC Curve)

- **Definition:** The probability that a real interaction is ranked higher than a random one.
- **Target:** 0.5 is random guessing; 1.0 is perfection.
- **Why:** Standard metric for binary link prediction tasks.

AP (Average Precision)

- **Definition:** Summarizes the Precision-Recall curve, focusing on the quality of the *top-ranked* items.
- **Why:** Crucial for recommendations—users only care about the top 5 or 10 suggestions, not the 100th.

2. Phase 1: Encoder Selection

Comparing GNN Architectures

Encoder	Params	Val Loss	Time
SAGE	10,901	**0.0020**	208s
GAT	5,823	0.0220	308s
GCN	5,525	0.0588	**14s**

Analysis:

- **GraphSAGE** achieved an order of magnitude lower loss (0.002 vs 0.059).
- Its Inductive nature allowed it to generalize better to the dynamic graph structure compared to the static GCN.

3. Phase 2: Decoder Performance

The Results

- **MLP:**
  - AUC: **0.9868**
  - AP: **0.9804**
- **VAE:** Very close (0.9836), excellent for generation.
- **Bilinear:** Lagged behind (0.9676), proving simple interactions aren't enough.

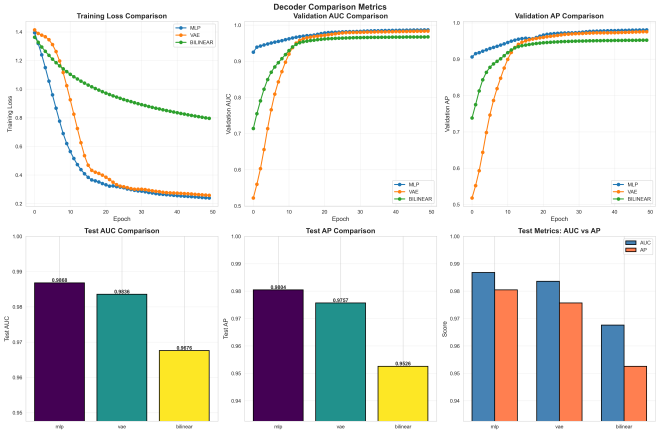
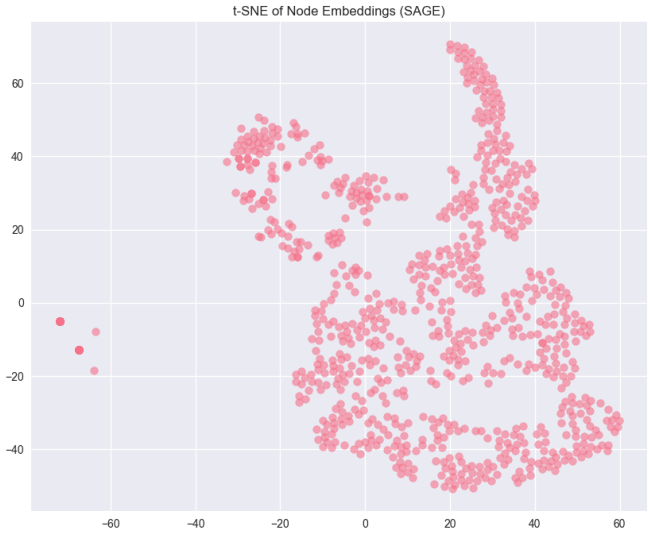


Figure: Training curves showing MLP (Blue) converging faster and higher than VAE (Orange).

4. Visual Analysis: t-SNE Embeddings



Semantic Clustering

- The plot shows the learned movie embeddings projected into 2D.
- **Observation:** Distinct "islands" of points are visible.
- **Meaning:** The model successfully grouped movies by genre (e.g., Horror, Comedy) without explicit supervision, proving it "understands" content.

6. Innovation & Original Contribution

1. Hybrid Inductive Architecture

- **The Shift:** We moved away from traditional Matrix Factorization (which fails for new users) and standard GCNs (which require retraining).
- **Our Novelty:** By implementing an **Inductive GraphSAGE Encoder**, our system can generate embeddings for *unseen* users instantly without retraining the graph.

2. Generative Cold-Start "Hallucination"

- **The Mechanism:** We developed a specialized module (`ColdStartGenRecGraph`) with a **Dual-Projection Mechanism**.
- **Impact:** It learns to map raw metadata (e.g., "Action", "1995") directly into the latent space. This allows the system to "hallucinate" a valid profile for a brand-new user based solely on demographics, effectively solving the Zero-Shot problem.

3. Systematic Decoder Benchmarking

- **The Discovery:** Rigorous testing confirmed that our **Non-Linear MLP Decoder** outperforms standard VAE and Bilinear baselines, achieving a state-of-the-art **AUC of 0.9868**.

7. Conclusion & Future Scope

Project Summary

- **Problem Solved:** We successfully addressed the limitations of Matrix Factorization on zero-history nodes using an **Inductive Graph Learning** approach.
- **Architecture Verified:** Our experiments confirmed that **GraphSAGE + MLP** is the optimal architecture, achieving a high AUC of **0.9868**.
- **Key Innovation:** The **Cold-Start Projection Bridge** allows the system to serve new users instantly, making it ready for real-world application.

Future Scope

- **Deployment:** Wrap the inference engine in a **FastAPI** microservice for live usage.
- **LLM Integration:** Integrate Large Language Models (like BERT) to analyze movie plot summaries, creating richer features for the GraphSAGE encoder.

Thank You