(An undertaking of Bhaktapur Municipality)

Khwopa College of Engineering

Affiliated to Tribhuvan University

Libali-08,Bhaktapur,Nepal

A

REPORT ON

"Movie Recommendation System"

IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE BACHELOR'S DEGREE IN COMPUTER ENGINEERING

Submitted by:

Saurab Khatiwoda (KCE077BCT034)

Suman Adhikari (KCE077BCT038)

Suraj Timilsina (KCE077BCT039)

Utshab Timalsina (KCE077BCT047)

Under the Supervision of

Er. Mukesh Kumar Pokharel

Department of Computer Engineering

Submitted to:

Department of Computer and Electronics Engineering

Khwopa College of Engineering

Bhaktapur,Nepal

2024-25

Copyright

Acknowledgement

Acknowledgement

We take this opportunity to express our deepest and sincere gratitude to our supervisor **Er. Mukesh Kumar Pokharel**, for his insightful advice, motivating suggestions, invaluable guidance, help, and support in the successful completion of this project, and also for his constant encouragement and advice throughout our Bachelor's program.
Additionally, we would like to thank our HoD **Er. Dinesh Gothe** and **Er. Sushil Dyopala** for providing valuable suggestions and for supporting the project.

Saurab Khatiwada	KCE077BCT034
Suman Adhikari	KCE077BCT038
Suraj Timilsina	KCE077BCT039
Utshab Timalsina	KCE077BCT047

Abstract

This paper presents a hybrid movie recommendation system designed to address the challenges of personalization and scalability in large-scale recommender systems. The proposed system integrates three distinct recommendation approaches: content-based, neural collaborative filtering (NeuMF), and a graph neural network (GNN). Content- based recommendations are derived from user profiles, constructed using a combination of explicit genre preferences and implicit features extracted from movie tags via RoBERTa embeddings. Collaborative filtering is achieved through a NeuMF model, leveraging user-movie interaction data. A GNN further enhances recommendations by capturing complex relationships within the user-movie interaction graph. The final recommendation is generated through a weighted average ensemble of the three models, optimizing for accuracy and diversity. To mitigate the cold-start problem for new users, the system incorporates an interactive questionnaire to elicit initial preferences, enabling immediate genre-based and popularity-based recommendations. A key feature of the system is its support for incremental learning; models are fine-tuned efficiently as new user ratings become available, avoiding computationally expensive full retraining. The system is implemented with a Django backend for data management, authentication, and API access, while the frontend is built using HTML, CSS, and React to provide a user-friendly interface for registration, login, rating submission, and recommendation retrieval. The entire system architecture is documented with detailed dot code, ensuring readability.

Keywords: *Recommender Systems, Hybrid Methods, Incremental Learning, Cold Start Problem, Collaborative Filtering, Content-Based Filtering, Graph Neural Networks, RoBERTa Embeddings*

Contents

List of Tables

List of Figures

List of Symbols and Abbreviation

ALS	Alternating Least Squares
API	Application Programming Interface
CAFF	Cross-Attention Feature Fusion
CF	Collaborative Filtering
CSV	Comma-Separated Values
DBN	Deep Belief Network
DCG	Discounted Cumulative Gain
DCG@K	Discounted Cumulative Gain at K
EM	Expectation Maximization
FCM	Fuzzy C-Means
GA	Genetic Algorithm
GNN	Graph Neural Network
IDCG@K	Ideal Discounted Cumulative Gain at K
IMDB	Internet Movie Database
IUD	Item-User Decoder
KNN	K-Nearest Neighbors
MAE	Mean Absolute Error
MAE@K	Mean Absolute Error at K
MBO	Monarch Butterfly Optimization
MSE	Mean Squared Error
NDCG@K	Normalized Discounted Cumulative Gain at K
NeuMF	Neural Matrix Factorization
Precision@K	Precision at K
Recall@K	Recall at K
RMSE	Root Mean Squared Error
RMSE@K	Root Mean Squared Error at K
RoBERTa	Robustly Optimized BERT Approach

ROC	Receiver Operating Characteristic
SGD	Stochastic Gradient Descent
SHAP	SHapley Additive exPlanations (interpretability metric)
SVD	Singular Value Decomposition
TF-IDF	Term Frequency-Inverse Document Frequency
TMDB	The Movie Database

CHAPTER 1

Introduction

1.1 Background

The increase in all spheres of digital content particularly and mostly beyond on-demand video streaming poses the acute challenge of increasingly discovering what users would like to view. Indeed, most traditional means of searching through keywords or sorts or genres of content cannot capture the uniquely specific connoisseurship. This project therefore develops high-quality recommendation systems to ameliorate the core problem of information overload pertaining to movies, leading to a situation better experienced by users, perfume surfing through what they would most likely like to watch.

This project is beyond simple recommendation techniques while implementing a hybrid approach. It brings together the potential of several advanced machine learning algorithms including content-based filtering, collaborative filtering, and graph-based techniques into a wholesome, solid approach. Utilizing the strengths of all of these methods, it seeks to overcome the shortcomings of any single method, thereby providing a more articulated and relevant set of suggestions; so that the recommendations shall be accurate. Furthermore, system is designed to be dynamic behavior-the system will learn from the users' interactions (ratings) and use the learning to adapt its recommendations.

It even treats the cold-start problem, that is the problem where new users are provided recommendations without a previous history of interaction to the system. This will be realized through an introductory preference elicitation mechanism (through genre preference and ratings on selected movies), through which the system will be able to generate useful recommendations even in the form of an empty user profile. In the end, this aims to develop a truly usable application in the real world that can show how just simple advanced machine learning techniques can be used to solve common and significant problems in today's digital age.

1.2 Motivation

The motivation for this project stems from the overwhelming abundance of content available on modern streaming platforms and the inherent difficulty users face in discovering movies that truly align with their individual tastes. Traditional search methods often fall short, relying on broad genre classifications or keyword searches that fail to capture the nuances of personal preference. This project aims to address this challenge by developing a sophisticated movie recommendation system that leverages the power of machine learning to deliver personalized and relevant suggestions, thereby enhancing the user experience and promoting content discovery.

The project intends to combine collaborative filtering and other methods through a hybrid approach. Content-based filtering analyzes movie or television program attributes such as genre and tags, while the collaborative filtering technique learns from user-movie interaction patterns. Adding a Graph Neural Network (GNN) would enable us to model complex relationships present within data, capturing subtle links between users and particular movies that are easily missed by traditional methods. With these inputs combined, we have a more robust and accurate recommendation engine that can adjust to individual user preferences and provide a wider range of relevant suggestions to users with very little interaction history (the "cold start" problem). The use of a weighted average ensemble allows for considerable flexibility in tuning the contribution of each underlying model.

Its improved structure and, primarily, how users will interact and discover finds will spur this project onward. In short, we are providing a personalized and intuitive recommendation experience. This effectively translates to less time spent looking for something to watch and more time enjoying great movie recommendations. A good recommendation engine not only helps the end-user but also the content provider to leverage useful insights into enhancing engagement and possibly promote films that users may not have thought of on their own. The project is, therefore, a practical mechanism to apply state-of-the-art machine learning techniques to circumvent a simple but profoundly affecting situation in this digital day and age.

1.3 Problem Statement

Current movie recommender systems, which include memory-based collaborative filtering methods like Wu et al.'s [1], and even model-based deep-learning systems, such as that based on the Convolutional Neural Network from Mu et al. [2], in the MovieLens dataset, still have some serious limitations. Whereas Mu et al. [2] achieved lower RMSE scores than traditional methods, giving some coverage to the cold-start problem, others remain. Collaborative approaches or even CNN ones may not be able to adequately grasp the richness of item metadata, whereas memory-based techniques are fundamentally limited by issues of scalability and sparsity. This project intends to combine both by moving on to a model-based recommendation engine that collaborates and utilizes content-based features along with state-of-the-art graph-aided neural networks for better accuracy, robustness, scalability, and adaptability to changing user preferences, thus improving user experience to levels beyond current systems.

1.4 Objective

The objective of this study is:

- To build a personalized movie recommendation system that effectively handles new users as well as old users envolving preferences and accuracy.

1.5 Scope of project

This project focuses on the development and implementation of a functional prototype for a hybrid movie recommendation system. The system's scope is defined as follows:

Data: The system utilizes three publicly available/scraped datasets:

- A movie metadata dataset providing information such as titles, genres, descriptions, and poster URLs.
- A user-movie ratings dataset containing explicit user feedback.
- A dataset of user-provided tags associated with movies.

Recommendation Techniques: The core of the system comprises three distinct recommendation algorithms:

- **Content-Based Filtering:** Leveraging movie metadata (genres) and pre-computed RoBERTa embeddings derived from movie tags to construct movie and user profiles.
- **Neural Collaborative Filtering (NeuMF):** A neural network-based approach to learn latent factors from user-movie interaction data.
- **Graph Neural Network (GNN):** Exploiting the graph structure of user-movie interactions to capture complex relationships.

Ensemble Method: A weighted average ensemble combines the predictions from the individual models to generate a final recommendation list.
Cold Start Mitigation: The system addresses the cold-start problem for new users by employing an initial questionnaire to elicit genre preferences and ratings for a curated set of example movies. This information is used to generate initial recommendations based on genre similarity and overall movie popularity.
Incremental Learning: The system is designed to adapt to new data dynamically. As users provide additional ratings, the models are incrementally updated, avoiding the need for computationally expensive full retraining.
User Interface: A user-friendly web interface, implemented using React-Vite, HTML, and CSS, provides functionalities for user registration, login, movie rating, and recommendation viewing.
Backend: A django-based backend API manages user authentication, data persistence (using SQLite), model interaction, and recommendation generation.
Evaluation: The system's performance is evaluated using standard recommendation metrics, including RMSE, MAE, Precision@K, Recall@K, and NDCG@K, calculated on a held-out test set during initial model training. Continuous evaluation during the incremental update phase is not within the scope of this project.

CHAPTER 2

LITERATURE REVIEW

The development of movie recommendation systems has become essential in the world of digital entertainment. By using methods like content-based filtering, collaborative filtering, and hybrid models, these systems can improve user satisfaction by making personalized movie recommendations. Content-based filtering makes movie recommendations that correspond with the user's previous tastes by taking into account factors like actors, directors, and genres. Conversely, collaborative filtering makes use of user ratings to spot trends and suggest movies that people who are similar to you might enjoy. Hybrid models integrate both strategies to reduce the drawbacks of each one separately and enhance recommendation precision. Furthermore, sophisticated techniques such as deep learning are utilized to comprehend intricate user preferences and interactions, thereby improving the recommendation system and guaranteeing more pertinent and captivating movie recommendations.

2.1 Movie Recommendation System Using Collaborative Filtering

The paper authored by Wu et al. [1] presents a movie recommendation system utilizing Collaborative Filtering (CF) techniques to improve recommendation accuracy and user satisfaction. The system employs User-based filtering using Pearson Correlation Similarity and Item-based filtering using Log Likelihood Similarity, applied separately. The user-based filtering approach calculates the similarity between users based on their rating patterns, while the item-based filtering method assesses the similarity between items.

2.2 Multimodal Movie Recommendation System Using Deep Learning

The research conducted by Mu et al. [2] presents a multimodal movie recommendation system utilizing deep learning with a Convolutional Neural Network (CNN) approach. The system processes the MovieLens dataset and applies deep-learning techniques to extract hidden features of movies and users. The model achieved RMSE scores of 0.9908 and 0.9096 on MovieLens 100K and 1M datasets, respectively, outperforming traditional recommendation approaches like User-CF, Item-CF, and SVD. It effectively addresses the cold-start problem and enhances recommendation accuracy.

2.3 Personalized Course Resource Recommendation Method Based on GEMRec

This study of Wang et al. [3] proposes GEMRec, a graph-enhanced multimodal recommendation method, designed to integrate text, video, and audio features for online educational resources. GEMRec employs a graph attention network and differentiable pooling for better knowledge graph modeling. The model improves interpretability through SHAP values and large language models. Experiments on the MOOCCubeX dataset show strong generalization, achieving Precision@10 = 0.267, Recall@10 = 0.265, and NDCG@10 = 0.297, surpassing traditional collaborative filtering techniques.

2.4 A Multimodal Graph Recommendation Method Based on Cross-Attention Fusion

The research conducted by Li et al. [4] proposed a multimodal graph recommendation method and analyzed the impact of key components like the Item-User Decoder (IUD) and Cross-Attention Feature Fusion (CAFF) on performance. Their results showed that removing IUD or CAFF significantly reduced Recall@20 and NDCG@20 across datasets. For example, in the Baby dataset, the full model achieved Recall@20: 0.1056 and NDCG@20: 0.0437, but without IUD, these dropped to 0.0563 and 0.0188, respectively. A similar trend was observed in the Sports dataset, where the full model reached Recall@20: 0.1124 and NDCG@20: 0.0523, but removing IUD reduced these values to 0.0612 and 0.0259. Additionally, they examined the effect of self-supervised learning loss weight (1) and found that 1 = 0.01 provided the best performance, while higher values negatively impacted recommendations. These findings highlight the importance of cross-attention fusion and careful hyperparameter selection in enhancing multimodal graph-based recommendation systems.

2.5 Movie Recommendation System Using Sentiment Analysis from Microblogging Data

The research conducted by Kumar et al. [5] explores a movie recommendation system that integrates collaborative filtering, content-based filtering, and sentiment analysis from microblogging data to enhance recommendation accuracy and user engagement. The hybrid system utilizes sentiment analysis of movie-related tweets to gauge public opinion and sentiment, combining this with traditional collaborative and content-based filtering techniques. The authors report that the average precision in Top-5 and Top-10 recommendations for sentiment similarity, the hybrid model, and the proposed model are 0.54 & 1.04, 1.86 & 3.31, and 2.54 & 4.97, respectively.

2.6 Personalized Real-Time Movie Recommendation System

The research conducted by Zhang et al. [6] presents a personalized real-time movie recommendation system that clusters users based on profile attributes using the K-means algorithm. The system creates a virtual opinion leader for each cluster by averaging user ratings within the cluster. The recommendation process utilizes the Weighted Slope One-VU method, which compares rating differences between item pairs as rated by these virtual leaders. This approach aims to enhance the accuracy and relevance of recommendations by effectively capturing the collective preferences of user clusters.

2.7 Content-Based Movie Recommendation System Using Ge- nre Correlation

The paper authored by Reddy et al. [7] proposes a content-based movie recommendation system that leverages genre correlation to generate personalized recommendations. The system employs content-based filtering techniques to analyze and correlate movie genres, aiming to recommend movies that align closely with user preferences based on genre similarity. By focusing on genre attributes, the system effectively captures user interests and delivers targeted movie suggestions.

2.8 Construction of Personalized Movie Recommendation Mo- del Relying on Recurrent Neural Network

The paper authored by Xie [8] presents a personalized movie recommendation model constructed using a Recurrent Neural Network (RNN) architecture. The model integrates collaborative filtering with matrix factorization techniques and adopts a voting-based recommendation approach. By leveraging RNN's ability to capture sequential patterns in user behavior and preferences, the model effectively combines collaborative signals and item features to generate personalized movie recommendations. The evaluation of the model yields a ROC AUC value of 0.90.

2.9 Content-Based Movie Recommendation System Using MBO with DBN

The research conducted by Sridhar et al. [9] proposes a content-based movie recommendation system that utilizes a hybrid model combining Monarch Butterfly Optimization (MBO) for feature selection and Deep Belief Network (DBN) for classification. The MBO algorithm aids in selecting relevant features from the movie dataset, while the DBN model leverages these features for accurate classification and recommendation generation. The performance evaluation of the system reveals a Mean Absolute Error (MAE) of 0.716, Root Mean Square Error (RMSE) of 0.915, precision of 97.35%, and recall of 96.6%.

2.10 A Movie Recommender System: MOVREC

The paper authored by Kumar et al. [10] introduces MOVREC, a movie recommender system that utilizes collaborative filtering with the k-means algorithm. By applying k-means clustering to group users with similar movie preferences, the system identifies patterns in user behavior and generates recommendations based on the preferences of similar user clusters.

2.11 Expectation Maximization and GA-based Movie Recommender System

The research conducted by Asha K N et al. [11] proposes a movie recommender system that integrates Expectation Maximization (EM) with Genetic Algorithm (GA) for enhanced recommendation accuracy. The system employs an EM-based clustering approach, combined with Principal Component Analysis (PCA) and GA, to identify latent user preferences and optimize recommendation generation. The evaluation of the system on Yahoo Movie and Netflix datasets yields Mean Absolute Errors (MAE) of 0.681 and 0.701 respectively, and Root Mean Square Errors (RMSE) of 1.303 and 1.309 respectively. These results indicate the effectiveness of the proposed approach in providing accurate movie recommendations, thereby improving user satisfaction and engagement with the recommendation system.

2.12 Hybrid Model for Movie Recommendation System Using Fireflies and Fuzzy C-Means

The paper authored by Kumar et al. [12] proposes a hybrid model for movie recommendation that combines collaborative filtering with the Firefly Algorithm and Fuzzy C-Means (FCM) algorithm. The collaborative filtering approach is enhanced by integrating the Firefly Algorithm and FCM algorithm to improve recommendation accuracy and relevance. The model's performance is evaluated using the Mean Absolute Error (MAE), resulting in a value of 0.60. This low MAE value suggests that the hybrid model effectively captures user preferences and generates accurate movie recommendations, thereby enhancing user satisfaction and engagement with the recommendation system.

2.13 Movie Recommendation System Using SVD and Cosine Similarity

The research conducted by Rajarajeswari et al. [13] proposes a movie recommendation system that utilizes a hybrid approach combining Singular Value Decomposition (SVD) with cosine similarity. By integrating these two techniques, the system aims to enhance recommendation accuracy and relevance. SVD is employed to decompose the user-item interaction matrix, capturing latent factors underlying user preferences, while cosine similarity measures the similarity between items based on their feature vectors. This hybrid method enables the system to generate personalized movie recommendations tailored to individual user tastes.

2.14 Movie Recommendation System Using Cosine Similarity and KNN

The paper authored by Singh et al. [14] proposes a movie recommendation system that employs content-based filtering techniques with cosine similarity and the K-Nearest Neighbors (KNN) algorithm. The system utilizes cosine similarity to measure the similarity between movies based on their feature vectors, such as genre, actors, and plot keywords. Additionally, the KNN algorithm is applied to identify the most similar movies to a given input, thereby generating personalized recommendations.

2.15 Comprehensive Analysis on Movie Recommendation System Employing Collaborative Filtering

The research conducted by Thakker et al. [15] conducts a comprehensive analysis on a movie recommendation system employing collaborative filtering as the primary technique. In their evaluation, the system achieves a Mean Absolute Error (MAE) of 0.929 and a Mean Squared Error (MSE) of 1.495. These metrics serve as indicators of the system's predictive accuracy, with lower values indicating better performance.

2.16 User Profile Correlation Based Similarity Algorithm in Movie Recommendation System

The paper authored by Widiyaningtyas et al. [16] proposes a movie recommendation system that employs the User Profile Correlation-based Similarity (UPCSim) algorithm, leveraging collaborative filtering with correlation-based similarity metrics. This algorithm focuses on calculating similarities between user profiles to generate personalized recommendations. In their evaluation, the system achieves a Mean Absolute Error (MAE) value of 0.745 and a Root Mean Square Error (RMSE) value of 0.939.

2.17 Intelligent Movie Recommendation System Through Gr- oup-Level Sentiment Analysis in Microblogs

The research conducted by Li et al. [17] conducts an intelligent movie recommendation system that utilizes group-level sentiment analysis in microblogs. The system employs sentiment analysis techniques to analyze user sentiments expressed in microblogs related to movies. By mining patterns in the sentiment data, the system identifies common sentiment trends and preferences shared by groups of users. These insights are then used to generate personalized movie recommendations tailored to the sentiment profiles of different user groups.

Table 2.1: Summary of Movie Recommendation Systems

SN	Title	Author/s	Methodology	Metrics
1	Movie Recommendation System Using Collaborative Filtering	Ching-Seh (Mike) Wu, Deepti Garg and Unnathi Bhandary	User-based filtering using PearsonCorrelationSimilarity and item-based filtering using LogLikelihood-Similarity (separately)	-
2	Multimodal Movie Recommendation System Using Deep Learning	Yongheng Mu, Yun Wu	Deep Learning, Feature Extraction (MovieLens), RMSE-based Evaluation	RMSE: 0.9908 (100K), 0.9096 (1M)
3	Research on Personalized Course Resource Recommendation Method Based on GEMRec	Enliang Wang, Zhixin Sun	Graph Attention Networks, Differentiable Pooling, SHAP-based Interpretability	Precision@10: 0.267, Recall@10: 0.265, NDCG@10: 0.297
4	A Multimodal Graph Recommendation Method Based on Cross-Attention Fusion	Kai Li, Long Xu, Cheng Zhu, Kunlun Zhang	Cross-Attention Fusion, Embedding of Item IDs + User Interactions, Dataset Performance Evaluation	Best performance at learning rate = 0.01 across all three datasets.

Table 2.2: Summary of Movie Recommendation Systems continued.

SN	Title	Author/s	Methodology	Metrics
5	Movie Recommendation System using Sentiment Analysis from Microblogging Data	Sudhanshu Kumar, Shirsendu S. Halder, Kanjar De, and Partha Pratim Roy	A Hybrid System-Combines collaborative and content-based filtering with sentiment analysis from movie tweet.	The average precision in Top-5 and Top-10 for sentiment similarity, hybrid, and proposed model are .54 & 1.04, 1.86 & 3.31, and 2.54 & 4.97, respectively.
6	Personalized Real-Time Movie Recommendation System: Practical Prototype and Evaluation	Jiang Zhang, Yufeng Wang, Zhiyuan Yuan and Qun Jin	Clustering users based on profile attributes using K-means and creating a virtual opinion leader for each cluster by averaging user ratings and using Weighted Slope One-VU method, which compares rating differences between item pairs as rated by virtual leaders.	-

Table 2.3: Summary of Movie Recommendation Systems continued.

SN	Title	Author/s	Methodology	Metrics
7	Content-Based Movie Recommendation System Using Genre Correlation	SRS Reddy, Sravani Nalluri, Subramanyam Kunisetti, S. Ashok and B. Venkatesh	Utilizes content-based filtering with genre correlation to generate recommendations.	-
8	Construction of Personalized Movie Recommendation Model Relying on Recurrent Neural Network	Qin Xie	Collaborative filtering with matrix factorization techniques and voting-based recommendation approach.	0.90 ROC AUC value
9	Content-Based Movie Recommendation System Using MBO with DBN	S. Sridhar, D. Dhanasekaran and G. Charlyn Pushpa Latha	A content based filtering method with hybrid model that combines Monarch Butterfly Optimization (MBO) for feature selection and Deep Belief Network (DBN) for classification.	MAE and RMSE are 0.716 and 0.915 and its precision and recall are 97.35% and 96.6% respectively.
10	A Movie Recommender System: MOVREC	Manoj Kumar, D.K. Yadav, Ankur Singh and Vijay Kr. Gupta	Collaborative filtering using k-means algorithm.	-

Table 2.4: Summary of Movie Recommendation Systems continued.

SN	Title	Author/s	Methodology	Metrics
11	Expectation Maximization and GA-based Movie Recommender System	Asha K N and R Raj Kumar	Expectation maximization based clustering approach with Principal Component Analysis and Genetic Algorithm (GA).	MAE of 0.681 and 0.701 and RMSE 1.303 and 1.309 for Yahoo Movie and Netflix dataset.
12	Hybrid Model for Movie Recommendation System Using Fireflies and Fuzzy C-Means	M. Sandeep Kumar and Prabhu J.	Collaborative filtering with Firefly Algorithm and FCM algorithm.	MAE value of 0.60.
13	Movie Recommendation System	S. Rajarajeswari, Sharat Naik, Shagun Srikant, M. K. Sai Prakash and Prarthana Uday	Hybrid method with SVD and cosine similarity.	-
14	Movie Recommendation System using Cosine Similarity and KNN	Ramni Harbir Singh, Sargam Maurya, Tanisha Tripathi, Tushar Narula and Gaurav Srivastav	Content-based filtering with cosine similarity and K-Nearest Neighbour algorithm.	-

Table 2.5: Summary of Movie Recommendation Systems continued.

SN	Title	Author/s	Methodology	Metrics
15	A comprehensive analysis on movie recommendation system employing collaborative filtering	Urvish Thakker, Ruhi Patel and Manan Shah	Collaborative filtering	MAE and MSE of 0.929, 1.495 respectively.
16	User profile correlation-based similarity (UPCSim) algorithm in movie recommendation system	Triyanna Widiyaningtyas, Indriana Hidayah and Teguh B. Adji	Collaborative filtering with correlation-based similarity.	MAE value of 0.745 and RMSE value of 0.939.
17	An intelligent movie recommendation system through group-level sentiment analysis in microblogs	Hui Li, Jiangtao Cui, Bingqing Shen, and Jianfeng Ma	Sentiment analysis and pattern mining techniques.	-

CHAPTER 3

THEORETICAL BACKGROUND

3.1 Theoretical Background

This recommendation system utilizes a hybrid approach, combining content-based filtering, neural collaborative filtering (NeuMF), and a graph neural network (GNN) to generate personalized movie recommendations. A cold-start mechanism addresses the challenge of new users.

3.1.1 Content-Based Filtering

Content-based filtering recommends items similar to those a user has previously preferred. Movie similarity is based on genre and tag-derived features.

Feature Representation: Each movie m is represented by a concatenated feature vector:

$$\mathbf{f}_m = [\mathbf{g}_m, \mathbf{r}_m] \tag{3.1}$$

where:

- **Genre Vector** ($\mathbf{g}_m$): A binary vector, $\mathbf{g}_m \in \{0,1\}^{|G|}$, indicating genre membership. $g_{m,i} = 1$ if movie m belongs to genre i, and 0 otherwise.
- **RoBERTa Embedding Vector** ($\mathbf{r}_m$): A dense vector, $\mathbf{r}_m \in \mathbb{R}^d$, derived from RoBERTa. Movie tags T_m are preprocessed to T'_m, then encoded:

$$\mathbf{H}_m = R(T'_m) = [\mathbf{h}_{m,1}, \ldots, \mathbf{h}_{m,L}], \quad \mathbf{H}_m \in \mathbb{R}^{L\times d} \tag{3.2}$$

  The embedding is the mean of the hidden states:

$$\mathbf{r}_m = \frac{1}{L}\sum_{i=1}^{L} \mathbf{h}_{m,i} \tag{3.3}$$

User Profile: A user profile $\mathbf{u}$ is the average of the feature vectors of positively rated movies (M_u^+):

$$\mathbf{u} = \frac{1}{|M_u^+|}\sum_{m\in M_u^+} \mathbf{f}_m \tag{3.4}$$

Prediction: A feed-forward neural network predicts the rating $\hat{y}_{u,m}$. The input is the concatenated user profile and movie feature vector, $[\mathbf{u}, \mathbf{f}_m]$. Hidden layers use ReLU activation. The output layer uses a linear activation. The loss function is Mean Squared Error (MSE):

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N} \sum_{(u,m)} (y_{u,m} - \hat{y}_{u,m})^2 \tag{3.5}$$

3.1.2 Neural Collaborative Filtering (NeuMF)

NeuMF combines Generalized Matrix Factorization (GMF) and a Multi-Layer Perceptron (MLP).

GMF Component:

- User u embedding: $\mathbf{p}_u \in \mathbb{R}^K$
- Movie m embedding: $\mathbf{q}_m \in \mathbb{R}^K$
- Interaction: $\hat{y}_{u,m}^{(\text{GMF})} = \mathbf{p}_u \odot \mathbf{q}_m$ (element-wise product)

MLP Component:

- User u embedding: $\mathbf{p}'_u \in \mathbb{R}^K$
- Movie m embedding: $\mathbf{q}'_m \in \mathbb{R}^K$
- Interaction: Embeddings are concatenated and passed through an MLP with ReLU activations:

$$\begin{aligned} \mathbf{z}^{(0)} &= [\mathbf{p}'_u, \mathbf{q}'_m] \\ \mathbf{z}^{(l)} &= \sigma\left(\mathbf{W}^{(l)}\mathbf{z}^{(l-1)} + \mathbf{b}^{(l)}\right), \quad \text{for layers } l = 1, \dots, L \\ \hat{y}_{u,m}^{(\text{MLP})} &= \mathbf{z}^{(L)} \end{aligned}$$

NeuMF (Combined):

$$\hat{y}_{u,m} = \sigma\left(\mathbf{h}^T[\hat{y}_{u,m}^{(\text{GMF})}, \hat{y}_{u,m}^{(\text{MLP})}] + b\right) \tag{3.6}$$

where σ is the sigmoid function, and $\mathbf{h}$ and b are the output layer's weight vector and bias.

Normalization: Before combining with other models, the output is normalized:

$$x' = \frac{x - \min(X)}{\max(X) - \min(X)} \tag{3.7}$$

Loss Function: Binary Cross-Entropy (BCE):

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N} \sum_{(u,m)} [y_{u,m} \log(\hat{y}_{u,m}) + (1 - y_{u,m}) \log(1 - \hat{y}_{u,m})] \tag{3.8}$$

3.1.3 Graph Neural Network (GNN)

The GNN models user-movie interactions as a bipartite graph.

Graph Structure: Nodes are users and movies. Edges represent ratings, with weights as normalized ratings.

Node Features:

- Movie nodes: Initialized with $\mathbf{f}_m$
- User nodes: Various initializations (zero vectors, learned embeddings, or genre profiles)

Graph Convolutional Network (GCN): Node embeddings are updated by aggregating neighbor information:

$$\mathbf{h}_v^{(l+1)} = \sigma \left(\mathbf{W}^{(l)} \sum_{u \in \mathcal{N}(v)} \frac{r_{u,v}}{\sqrt{d_u d_v}} \mathbf{h}_u^{(l)} \right) \tag{3.9}$$

where $\mathbf{h}_v^{(l)}$ is the embedding of node v at layer l, $\mathcal{N}(v)$ are the neighbors of v, $r_{u,v}$ is the rating, d_u and d_v are node degrees, $\mathbf{W}^{(l)}$ is a weight matrix, and σ is ReLU.

Final Prediction: After L layers, a linear layer or dot product of final node embeddings predicts the rating.

Loss Function: Mean Squared Error (MSE):

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N} \sum_{(u,m)} (y_{u,m} - \hat{y}_{u,m})^2 \tag{3.10}$$

3.1.4 Weighted Average Ensemble

The final prediction is a weighted average:

$$\hat{y}_{u,m}^{(\text{final})} = w_{\text{content}} \hat{y}_{u,m}^{(\text{content})} + w_{\text{NeuMF}} \hat{y}_{u,m}^{(\text{NeuMF})} + w_{\text{GNN}} \hat{y}_{u,m}^{(\text{GNN})} \tag{3.11}$$

where $w_{\text{content}} + w_{\text{NeuMF}} + w_{\text{GNN}} = 1$.

3.1.5 Cold Start Handling

For new users:

- **Genre Preferences:** Users select preferred genres. Movies are recommended based on cosine similarity:

$$\text{Similarity}(\mathbf{u}, \mathbf{g}_m) = \frac{\mathbf{u} \cdot \mathbf{g}_m}{\|\mathbf{u}\|\|\mathbf{g}_m\|} \tag{3.12}$$

- **Initial Ratings:** Users rate example movies to create an initial user profile.

- **Popularity-Based:** Recommends movies with the highest average rating:

$$\text{Popularity}(m) = \frac{1}{|R_m|} \sum_{r \in R_m} r \tag{3.13}$$

CHAPTER 4

Methodology

4.1 Software Development Approach

The Prototype model is a software development approach wherein instead of freezing the requirements before design or coding can proceed, a throwaway prototype is built to understand the requirements. The prototypes are usually incomplete systems, with many details left out. The goal is to provide a system with overall functionality. In this model, the prototype of the actual system is created, the requirements are updated, and the system is rebuilt until the final requirements are met.

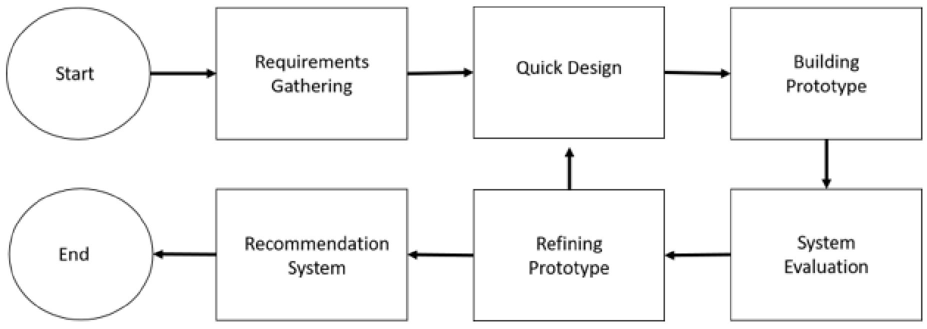


Figure 4.1: Prototype Model for Software Development

4.2 Datasets

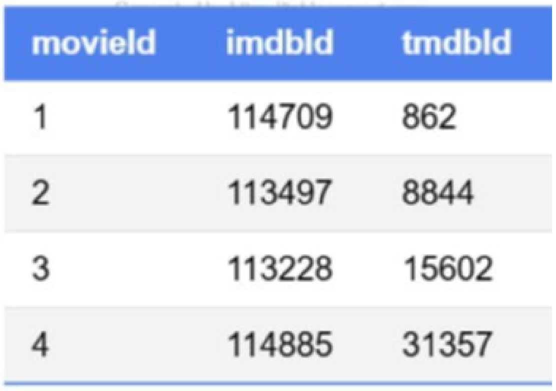
4.2.1 Data Collection and Preparation

MovieLens 32M Dataset

The MovieLens 32M dataset is a comprehensive resource for developing and benchmarking recommendation systems. It contains 32 million ratings and two million tag applications applied to 87,585 movies by 200,948 users. This dataset was collected in October 2023 and released in May 2024. It consists of four CSV files:

links.csv, *movies.csv*, *tags.csv*, and *ratings.csv*. Due to resource constraints, and training complexity, 8,75,000 ratings of 13,386 movies from 6502 users from this dataset are selected for the project. Below is a detailed explanation of each file and its purpose.

1. **links.csv**

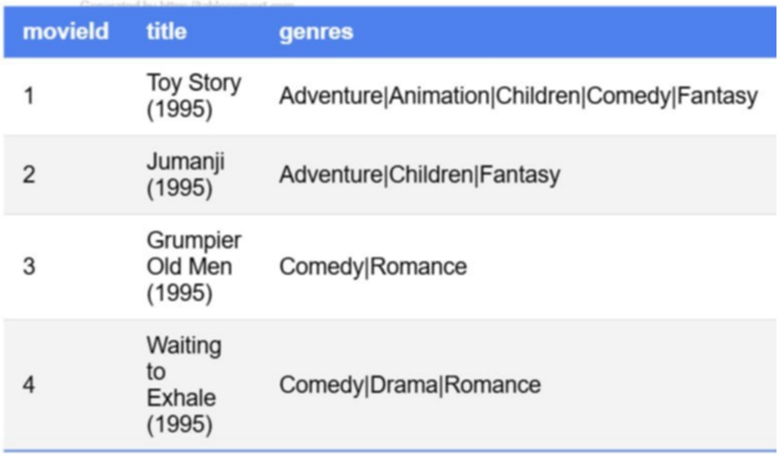


movieId	imdbId	tmdbId
1	114709	862
2	113497	8844
3	113228	15602
4	114885	31357

Figure 4.2: links.csv File Sample

- Purpose: This file maps MovieLens movie IDs to external movie database IDs, such as IMDb and TMDb.
- Columns:
  - movieId: A unique identifier for each movie in the MovieLens dataset.
  - imdbId: The corresponding ID for the movie on IMDb.
  - tmdbId: The corresponding ID for the movie on TMDb.

2. **movies.csv**

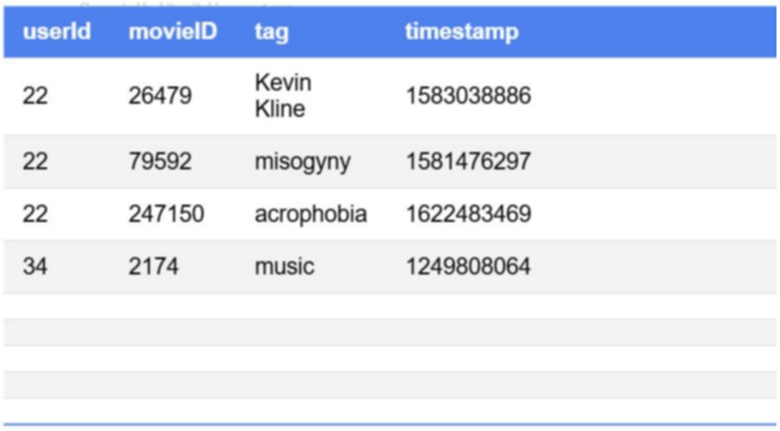


movieId	title	genres
1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	Jumanji (1995)	Adventure|Children|Fantasy
3	Grumpier Old Men (1995)	Comedy|Romance
4	Waiting to Exhale (1995)	Comedy|Drama|Romance

Figure 4.3: movies.csv File Sample

- Purpose: Provides metadata about movies, including titles and genres.
- Columns:
  - movieId: A unique identifier for each movie, matching entries in *links.csv* and *ratings.csv*.
  - title: The title of the movie (e.g., Inception (2010)).
  - genres: A pipe-separated (—) list of genres associated with the movie (e.g., Action—Thriller).

3. **tags.csv**



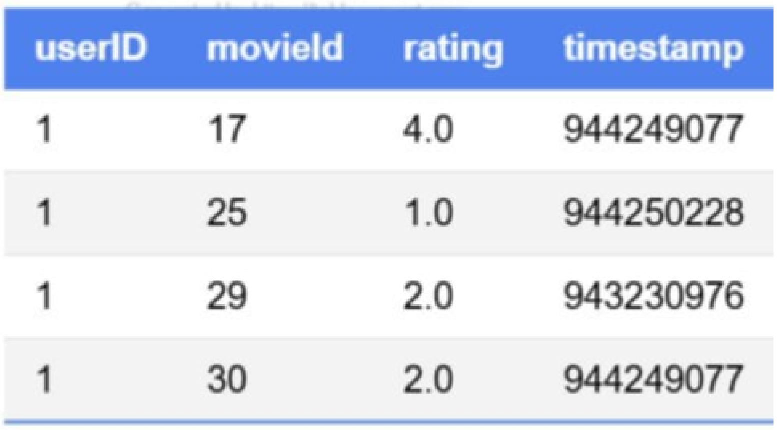
userId	movieID	tag	timestamp
22	26479	Kevin Kline	1583038886
22	79592	misogyny	1581476297
22	247150	acrophobia	1622483469
34	2174	music	1249808064

Figure 4.4: tags.csv File Sample

- Purpose: Provides metadata about movies, including titles and genres.

- Columns:
  - userId: A unique identifier for the user who applied the tag.
  - movieId: The ID of the tagged movie, matching entries in *movies.csv*.
  - tag: The descriptive tag applied by the user (e.g., mind-bending, classic).
  - timestamp: The time when the tag was applied, recorded as a Unix timestamp.

4. **ratings.csv**



userID	movieId	rating	timestamp
1	17	4.0	944249077
1	25	1.0	944250228
1	29	2.0	943230976
1	30	2.0	944249077

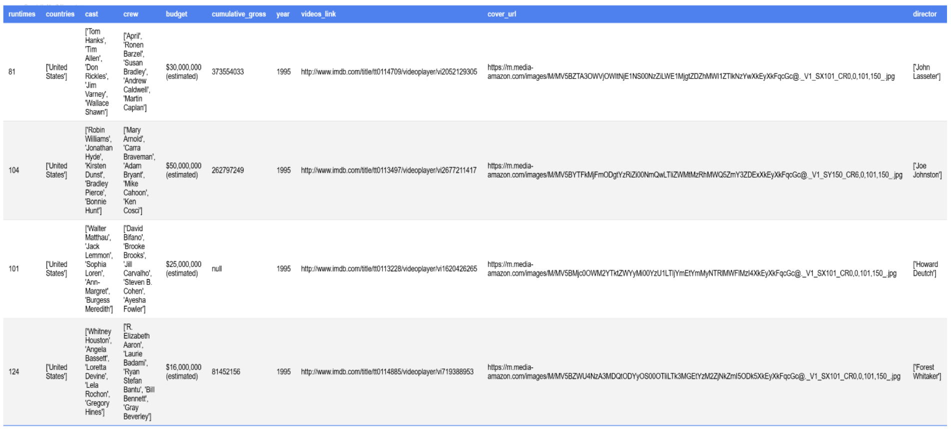
Figure 4.5: ratings.csv File Sample

- Purpose: Contains user ratings for movies, forming the core data for recommendation system development.
- Columns:
  - userId: A unique identifier for the user who rated the movie.
  - movieId: The ID of the rated movie, matching entries in *movies.csv*.
  - rating: The rating given by the user, typically on a scale from 0.5 to 5.0.
  - timestamp: The time when the rating was recorded, represented as a Unix timestamp.

4.2.2 Additional Metadata via API Integration

To enhance the dataset, we utilized the Cinemagor API with the help Movielens dataset (*links.csv*) using imdbIds to fetch additional movie details, including:

- **Language:** The primary language of the film.
- **Year:** The movie's release year.
- **Cover URL:** A direct link to the movie's poster or cover image.
- **Runtime:** The duration of the movie in minutes.
- **Countries:** The countries where the movie was produced.
- **Director:** The director(s) of the film.
- **Cast:** A list of 5 main actors featured in the movie.
- **Writers:** The screenplay or story writers list containing up to 5 writers.
- **Production Companies:** The studios or companies that produced the movie.
- **Budget:** The estimated production cost of the movie.
- **Cumulative Gross:** Total worldwide box office revenue.
- **Video link:** A link to the movie trailer.
- **Plot Outline:** Detailed information about the movie's storyline.
- **Plot:** A short explanation of the movie.
- **Full Size Cover URL:** A direct link to the movie's poster with a wider size.



runtimes	countries	cast	crew	budget	cumulative_gross	year	videos_link	cover_url	director
81	['United States']	['Tom Hanks', 'Tim Allen', 'Don Rickles', 'Jim Varney', 'Wallace Shawn']	['April', 'Ronen Barzel', 'Susan Bradley', 'Andrew Caldwell', 'Martin Caplan']	$30,000,000 (estimated)	373554033	1995	http://www.imdb.com/title/tt0114709/videoplayer/vi2052129305	https://m.media-amazon.com/images/M/MV5BZTA3OWVjOWItNjE1NS00NzZiLWE1MjgtZDZhMWI1ZTlkNzYwXkEyXkFqcGc@._V1_SX101_CR0,0,101,150_.jpg	['John Lasseter']
104	['United States']	['Robin Williams', 'Jonathan Hyde', 'Kirsten Dunst', 'Bradley Pierce', 'Bonnie Hunt']	['Mary Arnold', 'Carra Braveman', 'Adam Bryant', 'Mike Cahoon', 'Ken Cosci']	$50,000,000 (estimated)	262797249	1995	http://www.imdb.com/title/tt0113497/videoplayer/vi2677211417	https://m.media-amazon.com/images/M/MV5BYTFkMjFmODgtYzRiZi00NmQwLTliZWMtMzRhMWQ5ZmY3ZDExXkEyXkFqcGc@._V1_SY150_CR6,0,101,150_.jpg	['Joe Johnston']
101	['United States']	['Walter Matthau', 'Jack Lemmon', 'Sophia Loren', 'Ann-Margret', 'Burgess Meredith']	['David Bifano', 'Brooke Brooks', 'Jill Carvalho', 'Steven B. Cohen', 'Ayesha Fowler']	$25,000,000 (estimated)	null	1995	http://www.imdb.com/title/tt0113228/videoplayer/vi1620426265	https://m.media-amazon.com/images/M/MV5BMjc0OWM2YTktZWYyMi00YzU1LTljYmEtYmMyNTRlMWFlMzI4XkEyXkFqcGc@._V1_SX101_CR0,0,101,150_.jpg	['Howard Deutch']
124	['United States']	['Whitney Houston', 'Angela Bassett', 'Loretta Devine', 'Lela Rochon', 'Gregory Hines']	['R. Elizabeth Aaron', 'Laurie Badami', 'Ryan Stefan Bantu', 'Bill Bennett', 'Gray Beverley']	$16,000,000 (estimated)	81452156	1995	http://www.imdb.com/title/tt0114885/videoplayer/vi719388953	https://m.media-amazon.com/images/M/MV5BZWU4NzA3MDQtODYyOS00OTliLTk3MGEtYzM2ZjNkZmI5ODk5XkEyXkFqcGc@._V1_SX101_CR0,0,101,150_.jpg	['Forest Whitaker']

Figure 4.6: Enhanced Datasets File Sample

This enriched dataset provides a comprehensive view of each movie, enabling improved functionality for recommendation systems, detailed analysis, and insightful visualizations.

4.3 Proposed System Block Diagram

4.3.1 System Architecture

The Movie Recommendation System Architecture consists of several interconnected components that work together to generate movie recommendations. Users provide input through a User ID or Questionnaire, which helps in identifying whether they are new or existing users. For new users, a Cold Start Handling mechanism initializes recommendations based on available metadata. The system leverages Ratings Data, Tags Data, and Movies Data to build User-Movie Matrix and User Movie Features, which are essential for training models. The recommendation models include NeuMF (Neural Matrix Factorization), GNN (Graph Neural Networks), and Content-Based Filtering, which are combined using an Ensemble approach for better accuracy. The Update System continuously integrates new ratings from users to refine recommendations. The final output is personalized movie recommendations generated from the ensemble model, ensuring a dynamic and adaptive recommendation process.

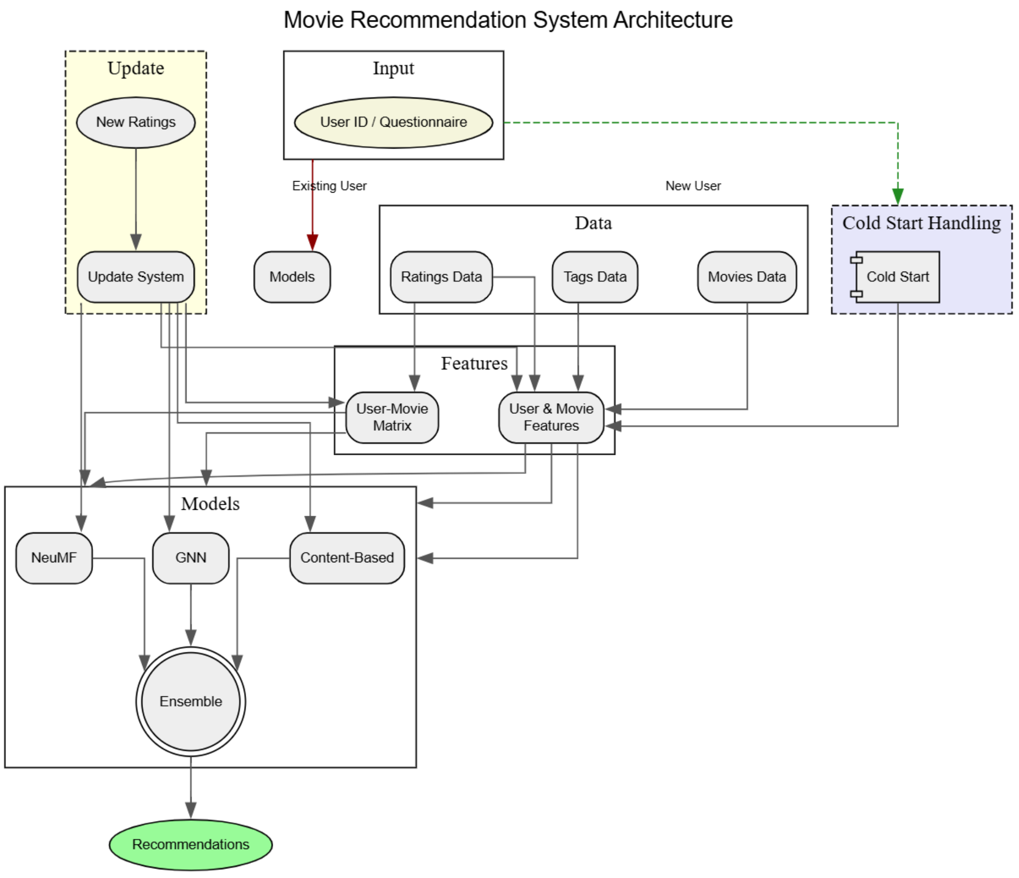


Figure 4.7: System Architecture

4.3.2 NeuMF (Neural Matrix Factorization)

The NeuMF model combines the strengths of Generalized Matrix Factorization (GMF) and a multilayer perceptron (MLP) to learn user-item interactions. It uses separate embeddings for users and movies in both the GMF and MLP components. The GMF component captures linear relationships, while the MLP component captures non-linear relationships through multiple dense layers. The outputs of the GMF and MLP components are concatenated and fed into a final dense layer to predict the rating. This approach seeks to capture both linear and non-linear patterns in the user-item interaction matrix. This seeks to get the interaction between the user and movie based on their rating data, rather than content.

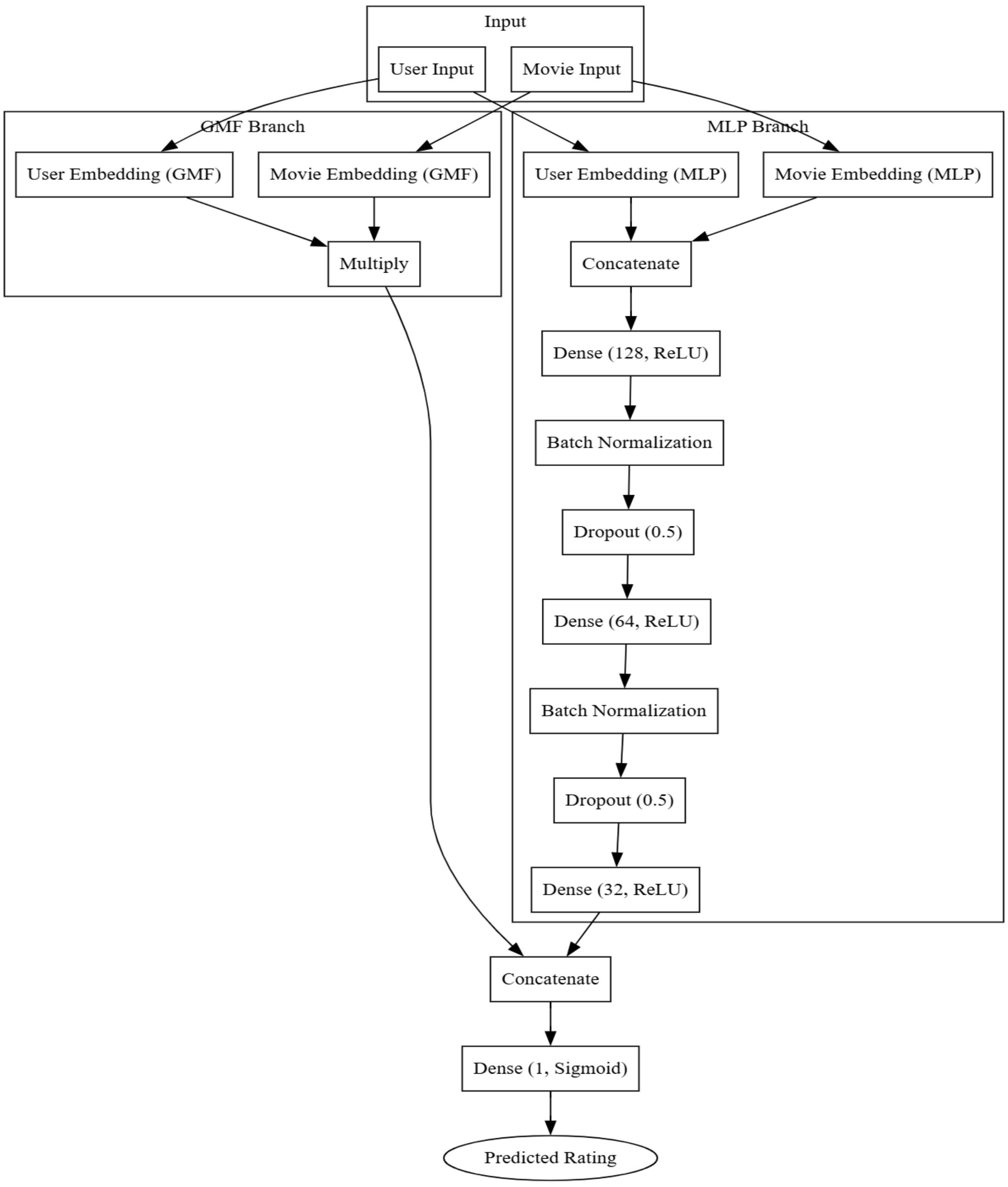


Figure 4.8: NeuMF Model Architecture

4.3.3 GNN (Graph Neural Network)

The GNN (Graph Neural Network) model represents the user-item interaction matrix as a graph, where users and movies are nodes, and ratings are edges. The model uses GCN (Graph Convolutional Network) layers to propagate information between neighboring nodes, learning node embeddings that capture the structure of the graph. The GNN architecture comprises multiple GCN layers followed by fully connected layers. These GNN networks learn through message passing between user and movie nodes, thereby capturing intricate relationships from user-movie interaction. After training the model is able to predict the rating a given user would give a movie, even with limited information.

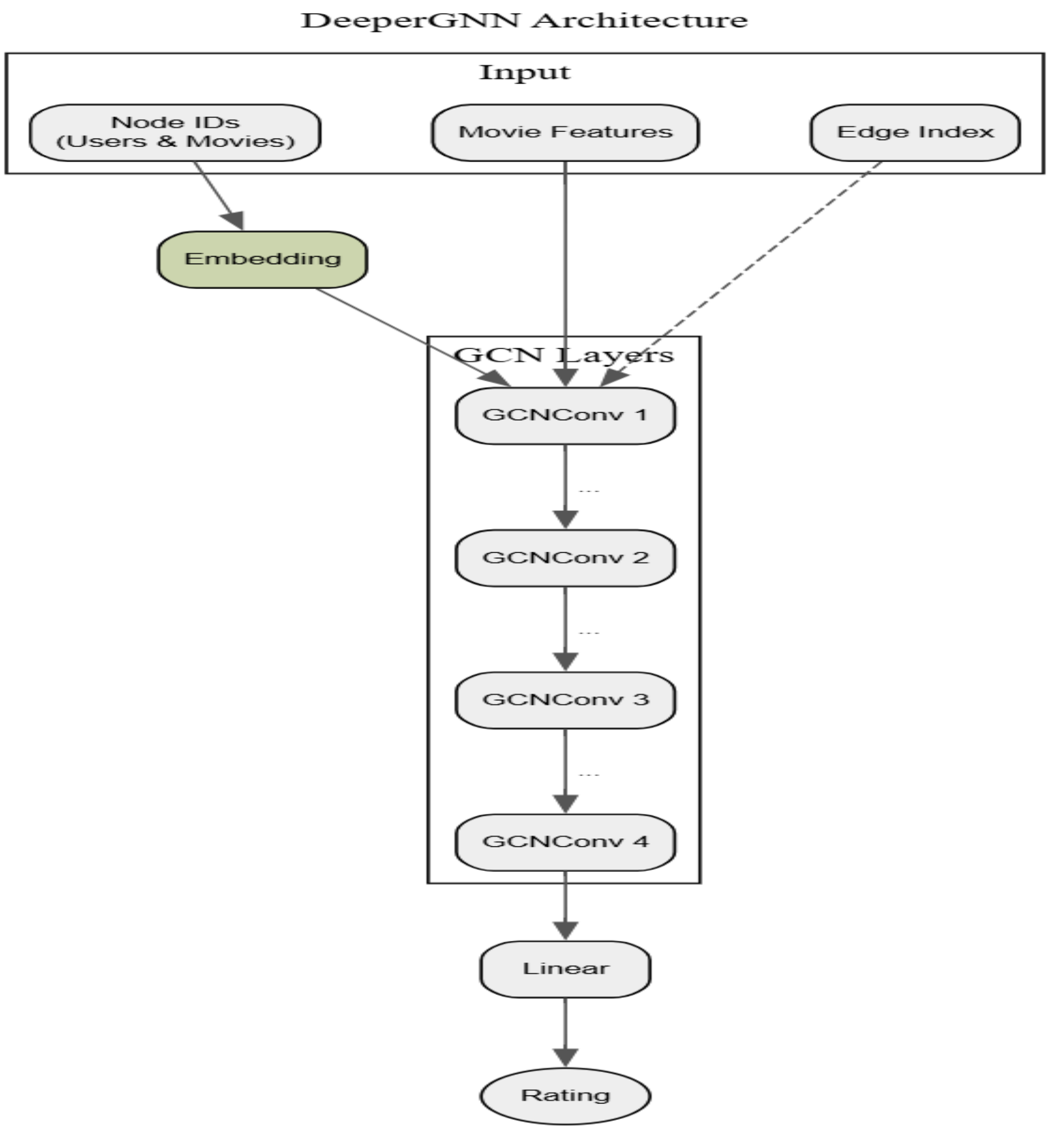


Figure 4.9: Deeper GNN Architecture

4.3.4 Content-Based Model

The content-based model utilizes user and movie features to predict ratings. It constructs user profiles based on their historical interactions with movie genres and tag embeddings generated by Roberta. These user profiles are then combined with movie features, including genres and RoBERTa-based tag embeddings, to create input vectors for the model. The model itself is a feedforward neural network trained to minimize the mean squared error between predicted and actual ratings. The purpose of this is to learn the underlying content, and it is a straightforward design with dense layers.

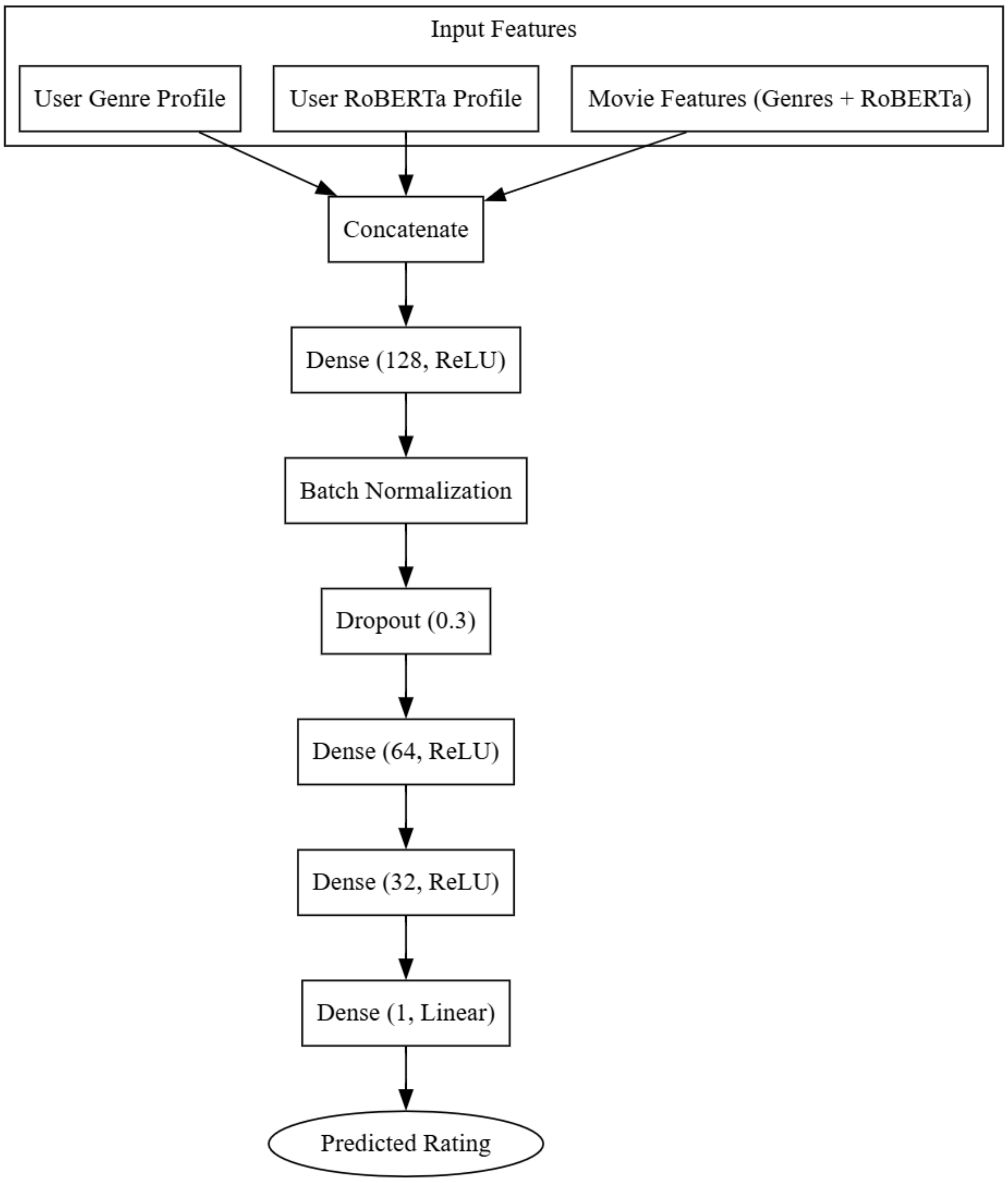


Figure 4.10: Content-Based Model Architecture

4.4 Description of Workflow and Pipeline of the System

4.4.1 Main Pipeline

The **Movie Recommendation Pipeline** follows a structured workflow consisting of four key stages: **Input, Processing, Output, and Update.**

1. **Input Stage:**The system begins with User Input, where users either provide an ID (for existing users) or complete a questionnaire (for new users). This step helps in identifying whether the user has prior interactions with the system or requires initialization.

2. **Processing Stage:**

   - If the user is new, the system applies Cold Start Handling, which uses metadata-based techniques to generate initial recommendations.
   - A User Profile is created or updated based on available data, including past interactions, preferences, or explicit inputs.
   - The system then applies Model Predictions using multiple approaches, such as Content-Based Filtering, Neural Matrix Factorization (NeuMF), and Graph Neural Networks (GNN).
   - These model predictions are then combined using a Weighted Average Ensemble approach to enhance accuracy and diversity in recommendations.

3. **Output Stage:** The final recommendations are generated in the form of a movie list, which is presented to the user.

4. **Update Stage:**

   - Users provide feedback by rating movies, which is processed through the Update System.
   - These new ratings are used to continuously refine the user profile and improve future recommendations.

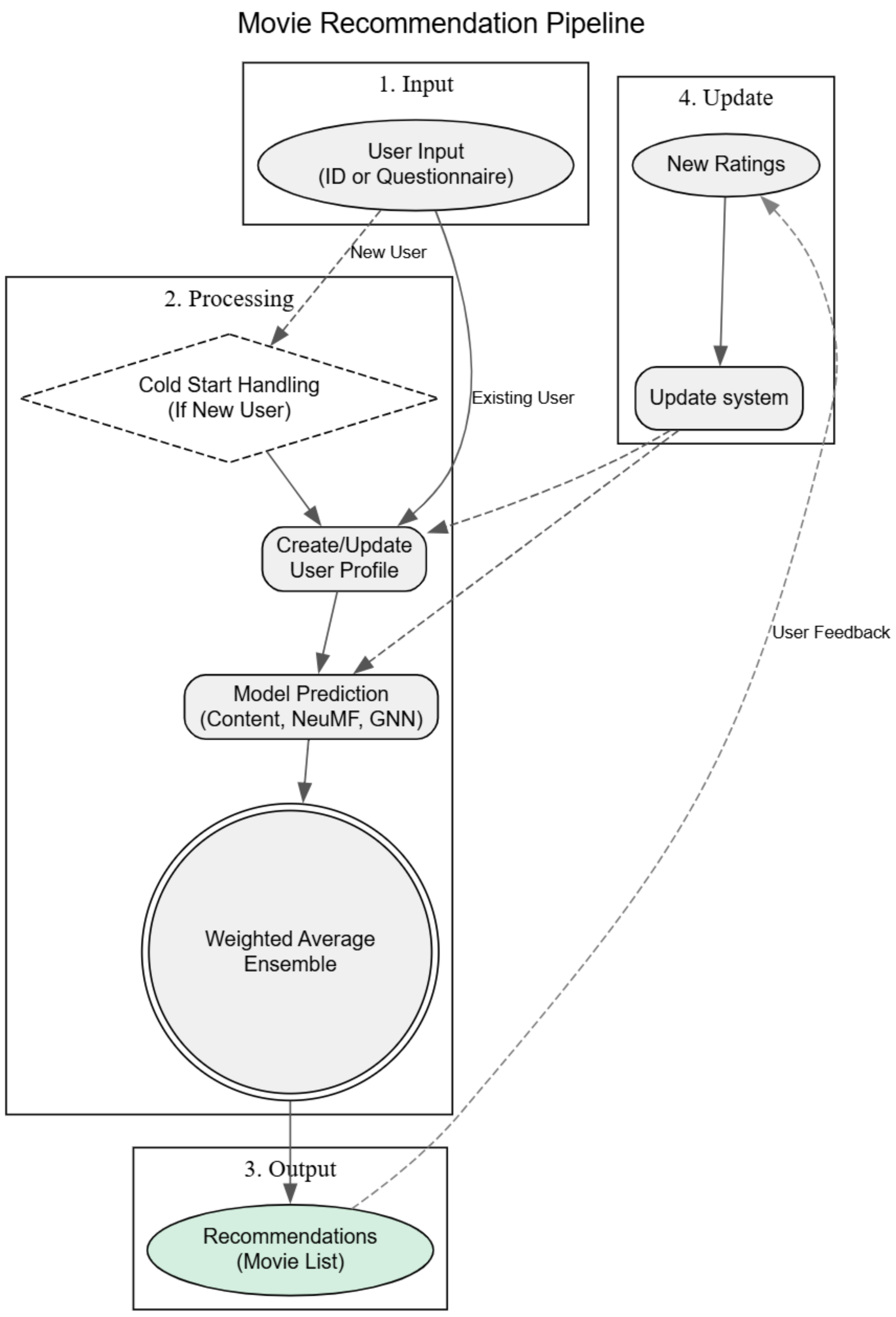

Movie Recommendation Pipeline
1. Input
User Input (ID or Questionnaire)
New User
Existing User
2. Processing
Cold Start Handling (If New User)
Create/Update User Profile
Model Prediction (Content, NeuMF, GNN)
Weighted Average Ensemble
3. Output
Recommendations (Movie List)
4. Update
New Ratings
Update system
User Feedback

Figure 4.11: System Pipeline

4.4.2 Cold Start Handling

The proposed architecture addresses the cold start problem in movie recommendation systems by utilizing a structured approach for new users who lack prior rating history. The system initiates the process by presenting a questionnaire, which collects user preferences based on genres and examples of movies. This step serves as an initial filtering mechanism to gauge user interests before making recommendations. The architecture then evaluates whether the user has provided sufficient ratings (at least three) on example movies to determine the appropriate recommendation strategy. If the user has rated at least three movies, the system generates rating-based recommendations, leveraging explicit feedback to refine suggestions. If the threshold is not met, the system resorts to genre-based recommendations, relying on the user's stated genre preferences. Additionally, pre-loaded data, including movie genres and popularity metrics, supports both genre-based and popularity-based recommendations to enhance accuracy. Ultimately, the system integrates these outputs to generate meaningful movie recommendations, ensuring that even new users receive relevant suggestions despite the absence of prior interactions.

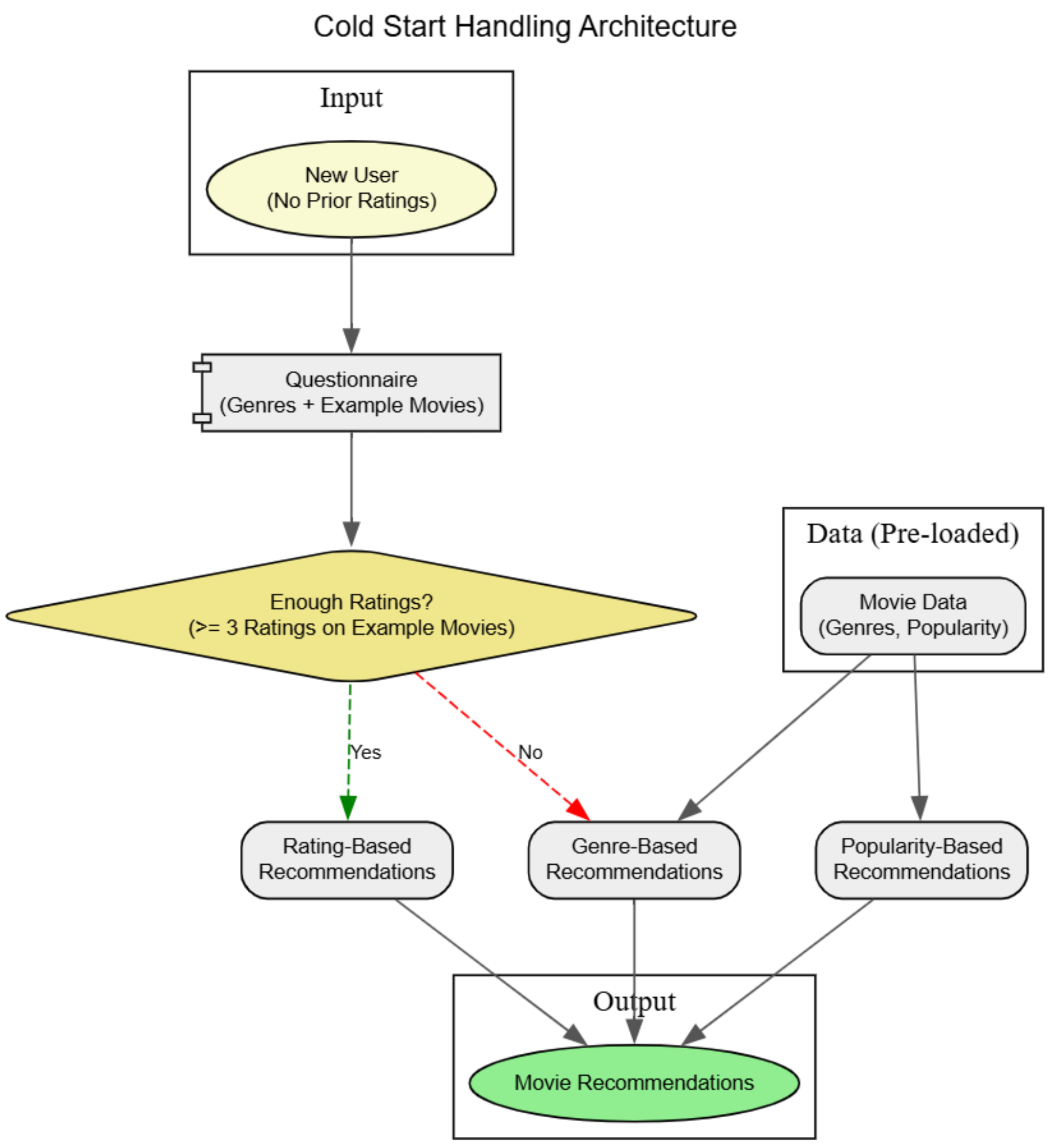

Cold Start Handling Architecture
Input
New User (No Prior Ratings)
Questionnaire (Genres + Example Movies)
Enough Ratings? (>= 3 Ratings on Example Movies)
Yes
No
Data (Pre-loaded)
Movie Data (Genres, Popularity)
Rating-Based Recommendations
Genre-Based Recommendations
Popularity-Based Recommendations
Output
Movie Recommendations

Figure 4.12: Cold Start Handling Architecture

4.4.3 Incremental Update

The Incremental Update Architecture is designed to continuously update a recommendation system as new user ratings become available. The process begins with the input of new user ratings, which include User ID, Movie ID, and Rating. These ratings are integrated into the current data, which consists of a User-Movie Matrix and User Profiles that store historical interactions and user-specific attributes. Upon receiving new ratings, the system proceeds with data updates, where the User-Movie Matrix is modified to include the latest ratings and User Profiles are recalculated to reflect updated user preferences. The updated data is then utilized for incremental training of multiple recommendation models, including GNN (Graph Neural Network), Content-Based filtering, and NeuMF (Neural Matrix Factorization). These models undergo retraining using the new data to improve recommendation accuracy. Finally, the trained models are stored as updated models, ensuring that the recommendation system remains adaptive and continuously learns from evolving user interactions.

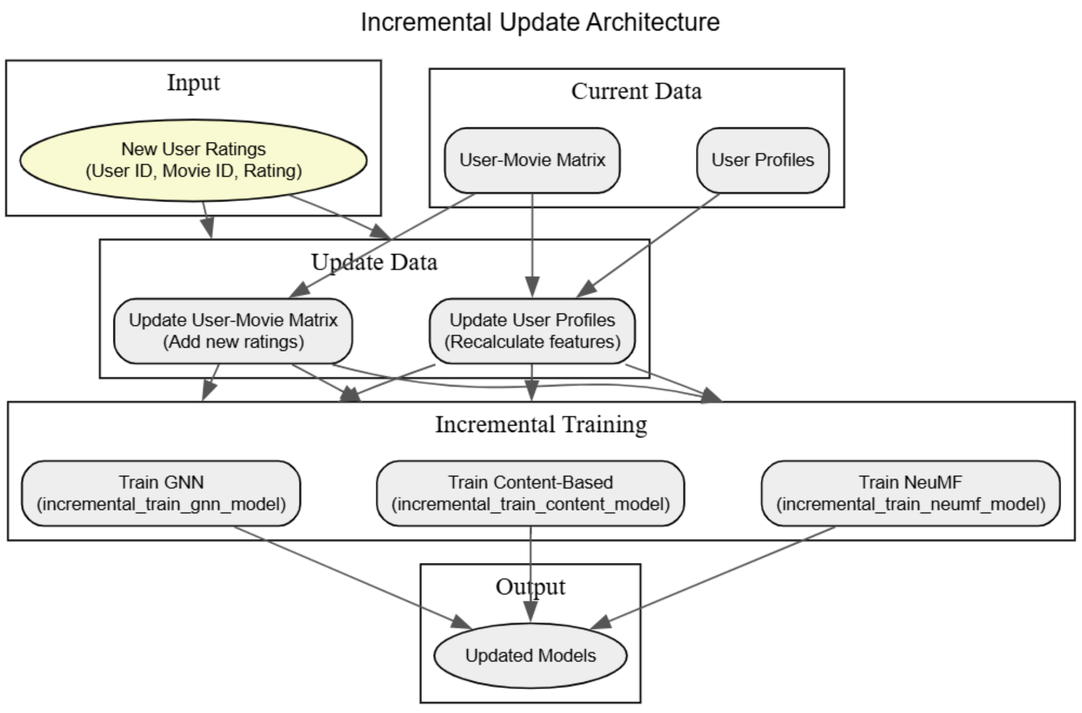

Incremental Update Architecture
Input
New User Ratings (User ID, Movie ID, Rating)
Current Data
User-Movie Matrix
User Profiles
Update Data
Update User-Movie Matrix (Add new ratings)
Update User Profiles (Recalculate features)
Incremental Training
Train GNN (incremental_train_gnn_model)
Train Content-Based (incremental_train_content_model)
Train NeuMF (incremental_train_neumf_model)
Output
Updated Models

Figure 4.13: Incremental Update Architecture

4.5 Performance Evaluation Metrics

To evaluate the performance of the movie recommendation system, the following are the performance metrics.

1. **RMSE (Root Mean Squared Error)**: Measures the average magnitude of the errors between predicted and actual ratings, giving more weight to larger errors.

   - **Formula**:

   $$RMSE = \sqrt{\frac{\sum (y_{\text{pred},i} - y_{\text{true},i})^2}{n}}$$

   - Where:
     - $y_{\text{pred},i}$ is the predicted rating for item i.
     - $y_{\text{true},i}$ is the actual rating for item i.
     - n is the total number of ratings.

2. **MAE (Mean Absolute Error)**: Measures the average magnitude of the errors between predicted and actual ratings, treating all errors equally.

   - **Formula**:

   $$MAE = \frac{\sum |y_{\text{pred},i} - y_{\text{true},i}|}{n}$$

   - Where variables are the same as RMSE.

3. **Precision@K**: Measures the proportion of recommended items in the top-K that are actually relevant (rated positively) by the user.

   - **Formula**:

   $$\text{Precision@K} = \frac{\text{Number of relevant items in top-K}}{K}$$

   - Where relevant items are true positives.

4. **Recall@K**: Measures the proportion of all relevant items for a user that are successfully retrieved in the top-K recommendations.

   - **Formula**:

   $$\text{Recall@K} = \frac{\text{Number of relevant items in top-K}}{\text{Total number of relevant items}}$$

5. **NDCG@K (Normalized Discounted Cumulative Gain)**: Measures the ranking quality of the recommendations, giving higher scores to relevant items ranked higher.

   - **Formula**:
$$\text{NDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}}$$
   - Where:
     - **Discounted Cumulative Gain (DCG@K)**:
$$\text{DCG@K} = \sum_{i=1}^{K} \frac{\text{rel}_i}{\log_2(i+1)}$$
     - rel_i is the relevance (e.g., 1 or 0, or the rating) of the item at rank i.
     - **Ideal DCG (IDCG@K)** is calculated by ranking the actual relevant items in the best possible order.

CHAPTER 5

System Design

5.1 Requirement Specification

5.1.1 Functional Requirements

1. **User Accounts:**
    - Users can register for an account.
    - Users can log in and log out.
2. **Movies:**
    - Users can see a list of movies.
    - Users can view details about each movie.
3. **Ratings:**
    - Logged-in users can rate movies.
4. **Recommendations:**
    - The system provides personalized movie recommendations to logged-in users.
    - Recommendations are based on a combination of different recommendation methods.
5. **New Users (Cold Start):**
    - New users without ratings can get recommendations based on:
        - Their chosen genres.
        - Ratings they give to a few example movies.
        - General movie popularity.
6. **System Updates:**
    - The system updates its recommendations when new ratings are added.

5.1.2 Non-Functional Requirements

1. Performance

    - The system must deliver recommendations quickly, with a response time of less than 2 seconds for generating personalized recommendations.

2. Scalability

    - The system must handle a large number of users and a growing database of movies without performance degradation.

3. Security

    - User data must be protected with appropriate security measures, including encryption and secure authentication mechanisms.

4. Availability

    - The system must be available 99.9% of the time, ensuring minimal downtime.

5. Usability

    - The user interface must be intuitive and easy to navigate, providing a seamless user experience.

6. Maintainability

    - The system must be designed for easy maintenance and updates, with modular components and clear documentation.

7. Data Privacy

    - The system must comply with data privacy regulations, ensuring that user data is collected, stored, and used in accordance with legal requirements.

8. Compatibility

- The system must be compatible with various devices and browsers, ensuring accessibility for all users.

9. Robustness

   - The system must handle errors gracefully, providing meaningful error messages and ensuring that user data is not lost during failures.

10. Adaptability

    - The recommendation engine must be adaptable to incorporate new algorithms and techniques as they become available.

5.2 Feasibility Assessment

5.2.1 Technical Feasibility

The project is considered technically feasible. The core technologies involved, namely, Python, Django REST Framework, React, and the various machine learning libraries (TensorFlow, PyTorch, scikit-learn, transformers, torch-geometric) required are all mature, well-documented, and widely used. Sufficient online resources, with tutorials and community support for these technologies, are available. The algorithms covered in this context (content-based filtering, neural collaborative filtering, graph neural networks, weighted averaging) are all well-established ones in the field of recommender systems. Further, the pre-trained models used (RoBERTa, and the initial training of the recommendation models) simplify model development, consequently reducing overhead. The incremental update method employed here is difficult to implement but widely practiced, thus eliminating the need for regular full model retraining, which is expensive with time and computation. The use of a relational database (SQLite) means that data integrity is assured, and a well-defined API is used for efficient communication between the frontend and backend. The availability of clouds for deployment and scaling (AWS, Google Cloud, Azure, etc.) is another plus.

5.2.2 Economic Feasibility

The project is very economically feasible. The main development tools and libraries (Python, Django, React, TensorFlow, PyTorch, etc.) are open-source and free to use, meaning that there will be no licensing costs. Development costs would mostly come from developer time. For a small deployment, hosting would be nearly zero using free tiers or low-cost options on cloud infrastructures. For large deployments, the infrastructure would be costed (e.g., compute, storage, database) by usage, still keeping this favorable for the potential with user engagement and content discovery. Also, the incremental update mechanism provides economic viability by saving costly computing sessions of retraining of the model. The open-source nature of the core technologies helps keep future maintenance and development open, thus lowering future costs. The whole project thus presents a favorable ROI, considering that it may enhance user satisfaction and content consumption.

5.2.3 Operational Feasibility

The operational feasibility is highly rated for this system. The architecture is chosen so that a clear separation exists between the frontend (React) and the backend (Django), thus supporting maintainability and allowing for independent updates and scaling of each component. The usage of the REST API provides clear communication among the frontend and backend, thus making debugging and modification of the system much simpler. The incremental update mechanism keeps the system alive with user ratings and updated without downtime.

5.3 Use Case Diagram

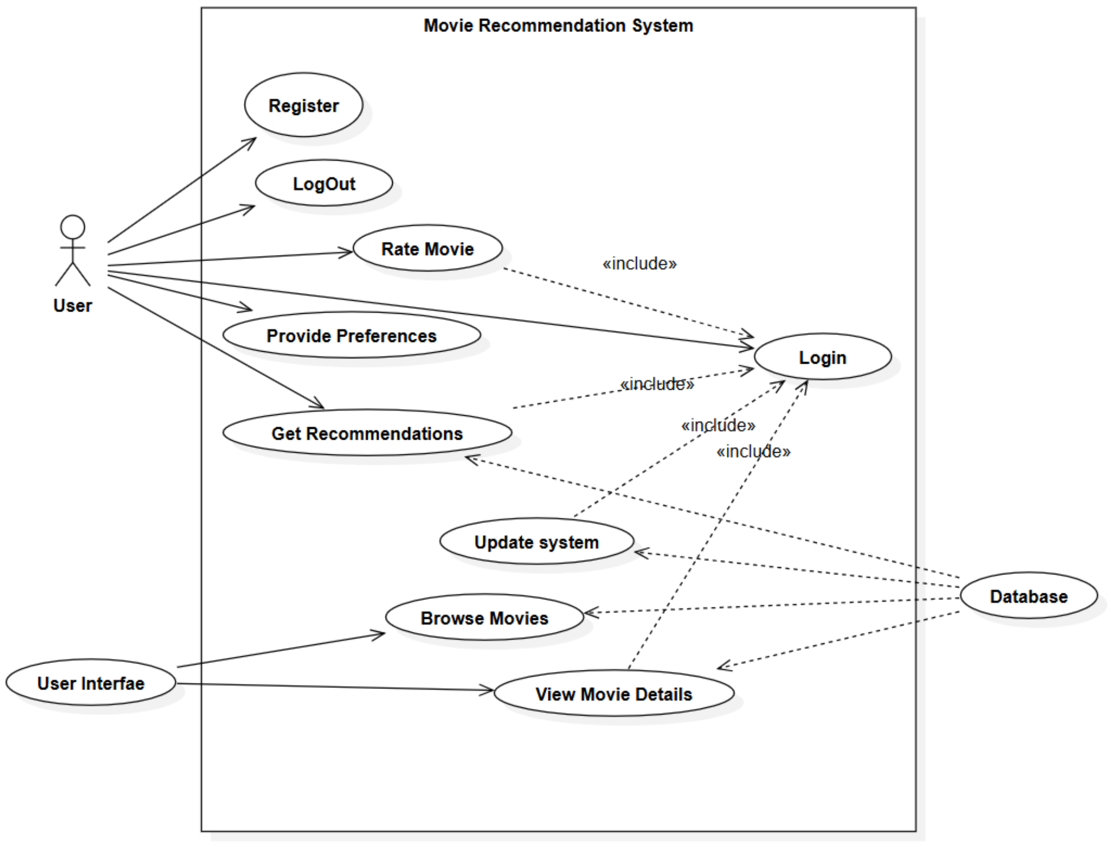


Figure 5.1: Use Case Diagram for Movie Recommendation System

5.4 Sequence Diagram

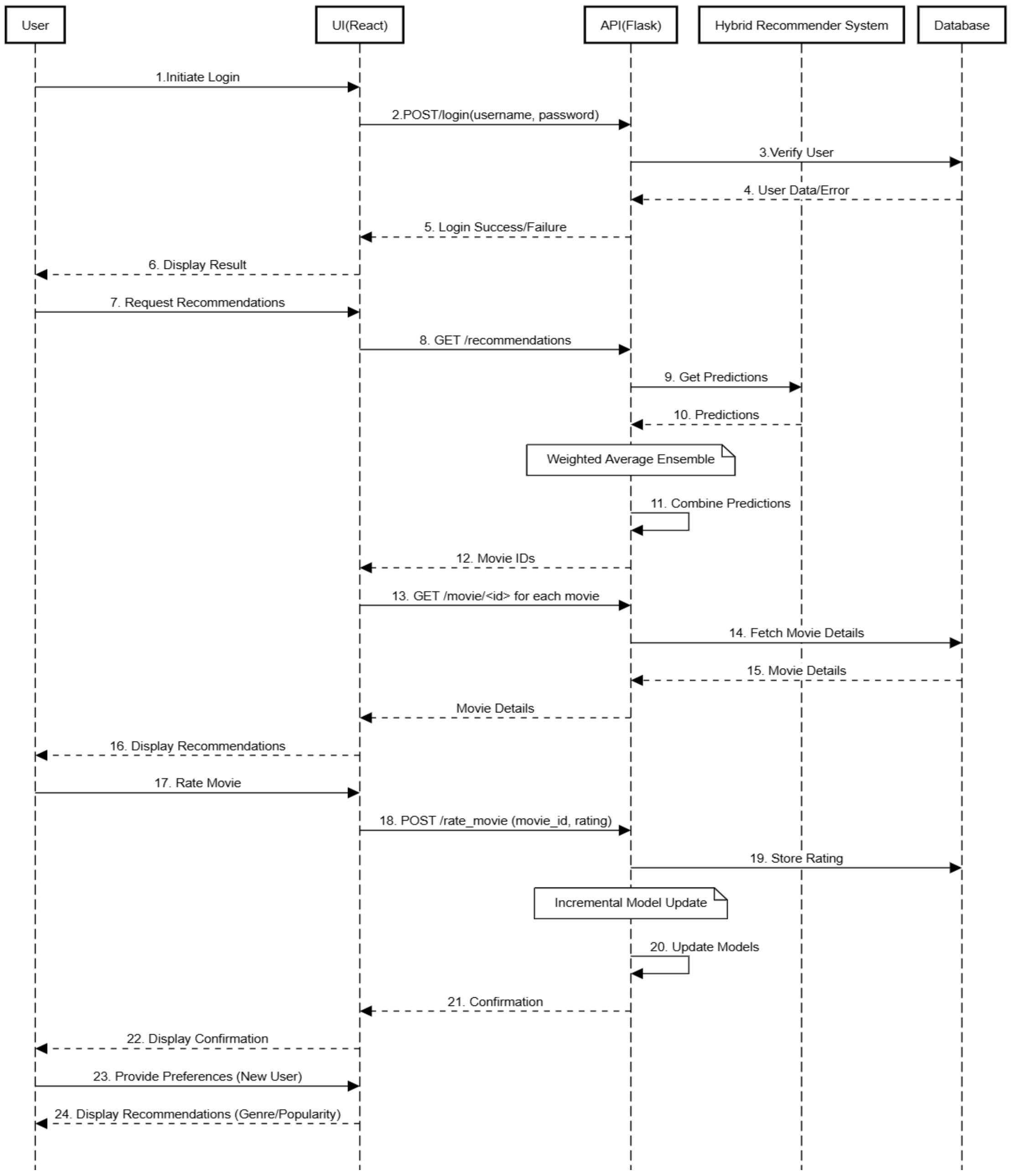


Figure 5.2: Sequence Diagram of the Movie Recommendation System

CHAPTER 6

Result and Discussion

6.1 Data Analysis

The primary goals of this analysis are to understand the distribution of ratings, identify popular genres, and explore trends in movie ratings over time and across users. This information provides valuable insights into user preferences and the overall characteristics of the movie rating dataset.

1. **Number of Movies per Genre**

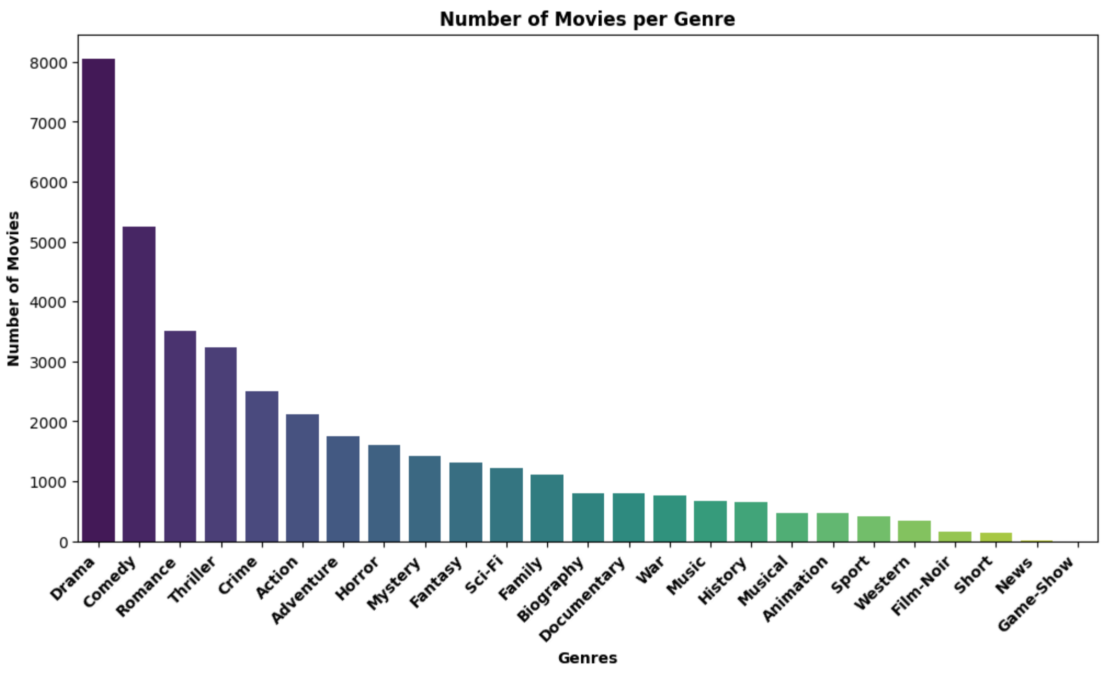


Figure 6.1: Number of Movies per Genre

The bar plot above shows the total number of movies per genre. This visual provides a general overview of the genre landscape within the dataset. There are 25 unique genres in the dataset.

2. **Top 10 Genres**

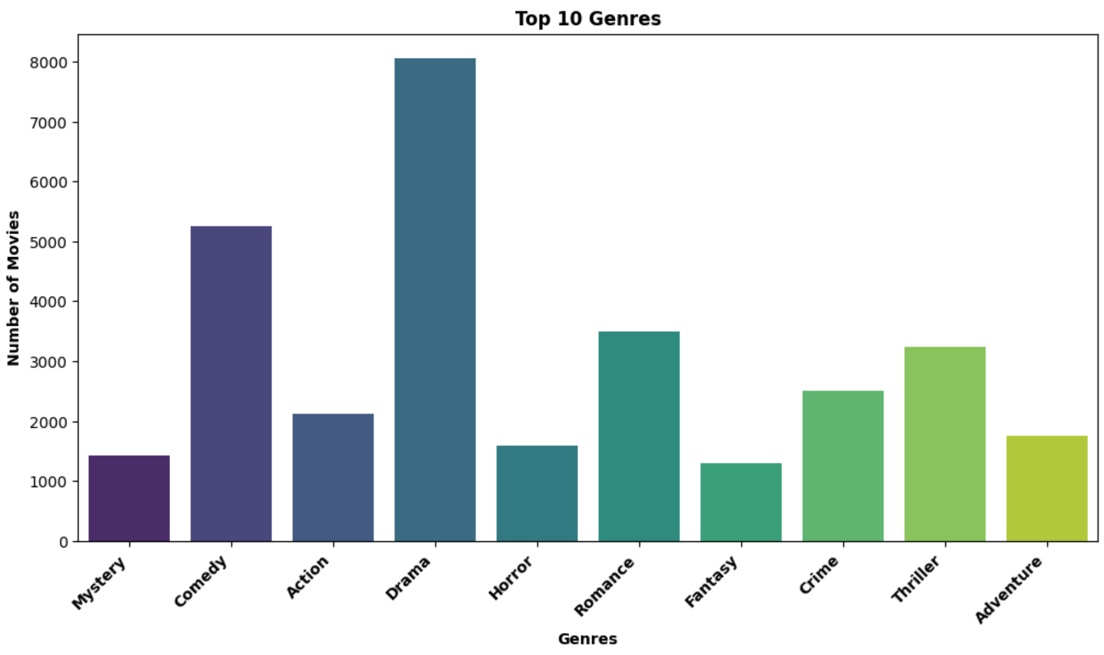


Figure 6.2: Top 10 Genre

The second visualization focuses on the top 10 most-rated movies in the dataset. This visualization helps to identify those movie genres that were rated most, implying that those genres were preferred more. Hence, the 10 top genres are Mystery, Comedy, Action, Drama, Horror, Romance, Fantasy, Crime, Thriller and Adventure.

3. **Most Rated Movies**

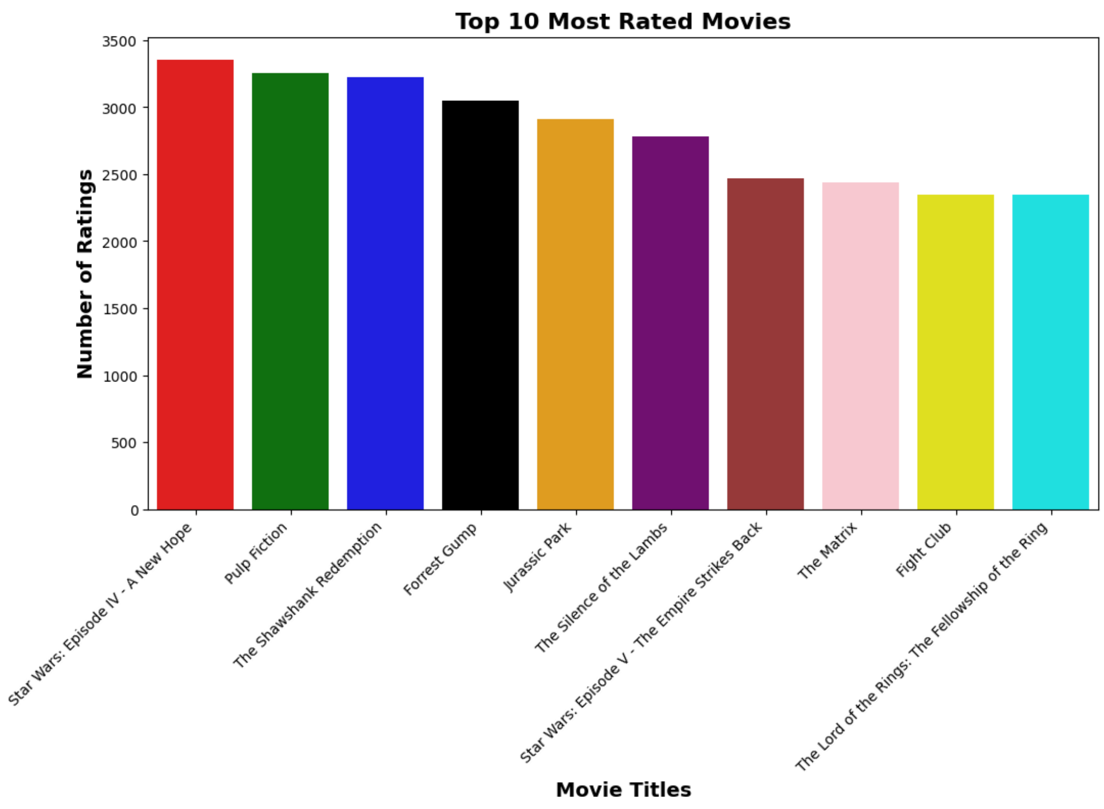


Figure 6.3: Top 10 Most Rated Movies

This graph highlights the most popular movies within the dataset, as determined by the number of ratings they have received. It looks like user have mostly rated Star Wars: Episode IV - A New Hope.

4. **Top Movie Raters**

The bar plot identifies the top 10 users who rated the most movies. Identifying these prolific raters can be useful for understanding their influence on the overall rating distribution. User number 28 has rated the most movies, with more than 2,500 ratings.

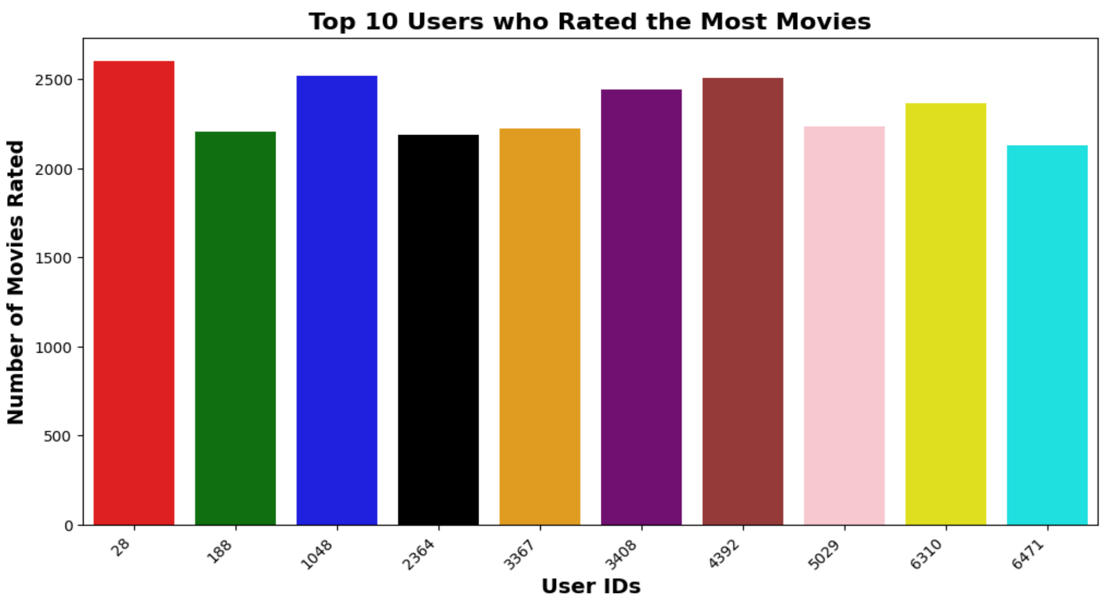


Figure 6.4: Top 10 users Who Rated the Most Movies

.

5. **Distribution of Ratings**

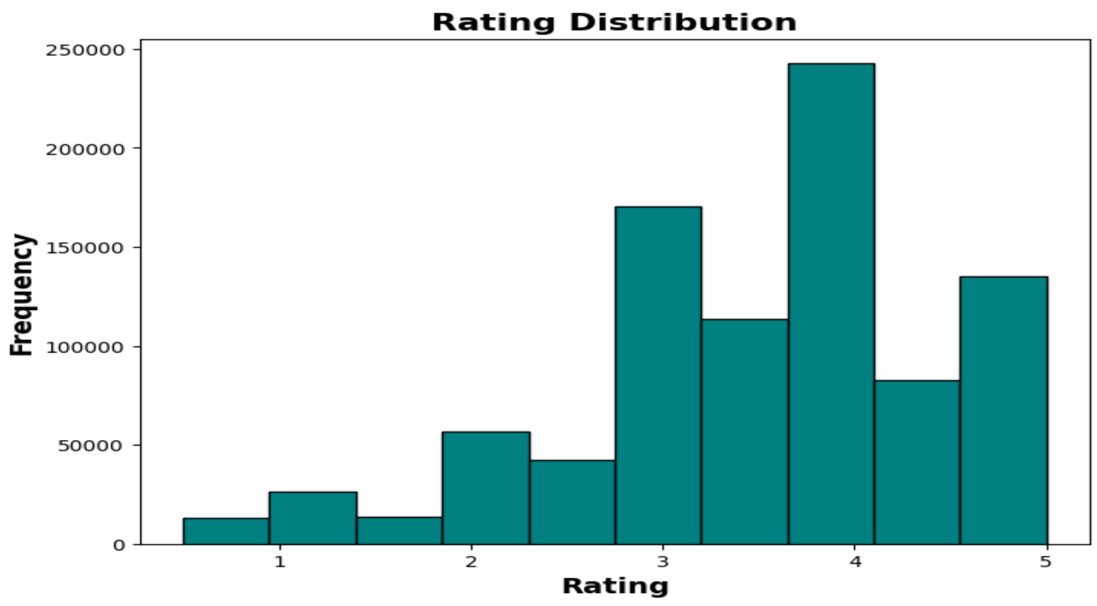


Figure 6.5: Rating Distribution Graph

The ratings distribution was explored using a histogram and a pie chart. The histogram provides a visual representation of the frequency of each rating value, revealing the overall shape of the rating distribution.

The pie chart provides a percentage distribution of the rating values.

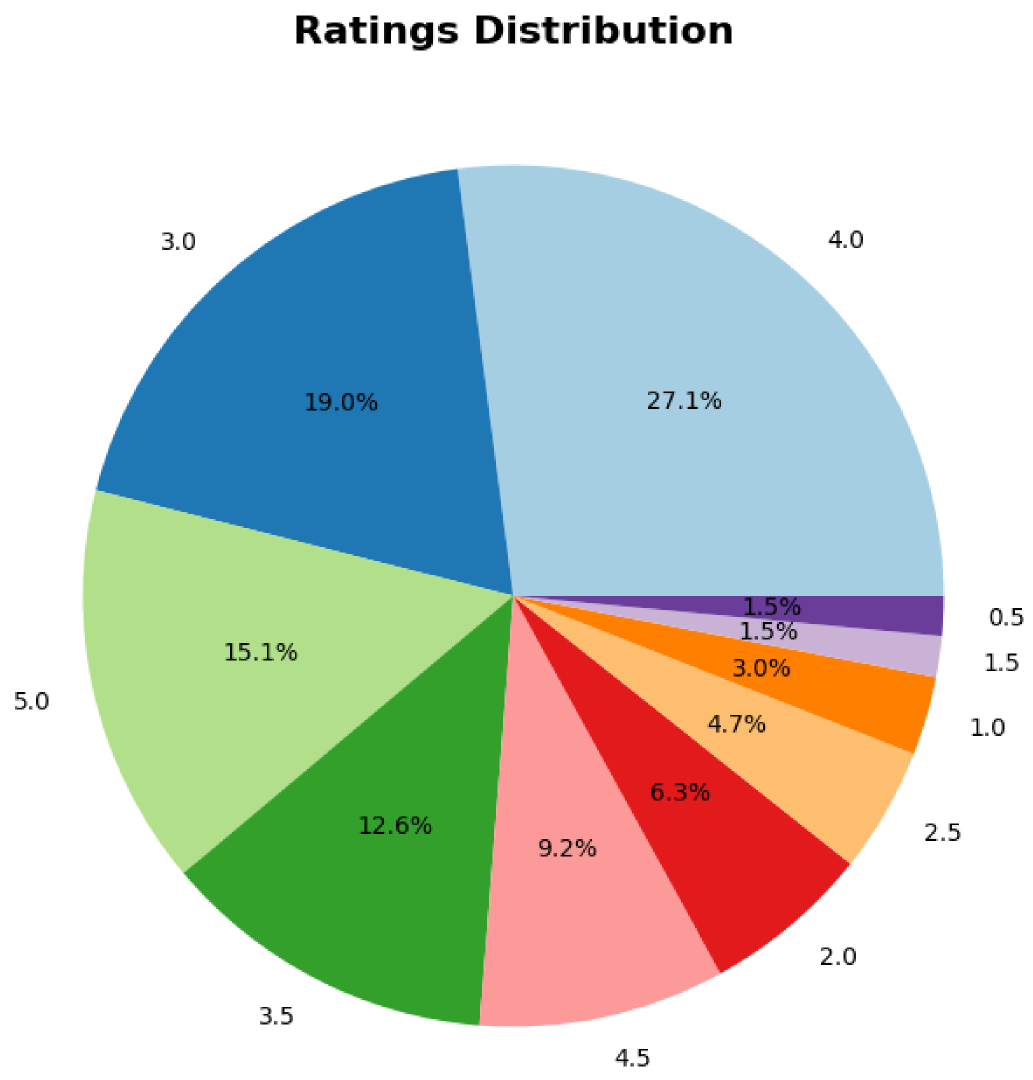


Figure 6.6: Rating Distribution Pie-Chart

From both illustrations, it appears that most users have rated 4 to the highest number of movies. Additionally, most of the movies in the dataset have received ratings from 3-5.

6. **Average Ratings Over Time**

   To investigate rating trends over time, a line plot was created showing the average rating per year. This visualization can reveal whether average ratings have increased, decreased, or remained stable over the years.

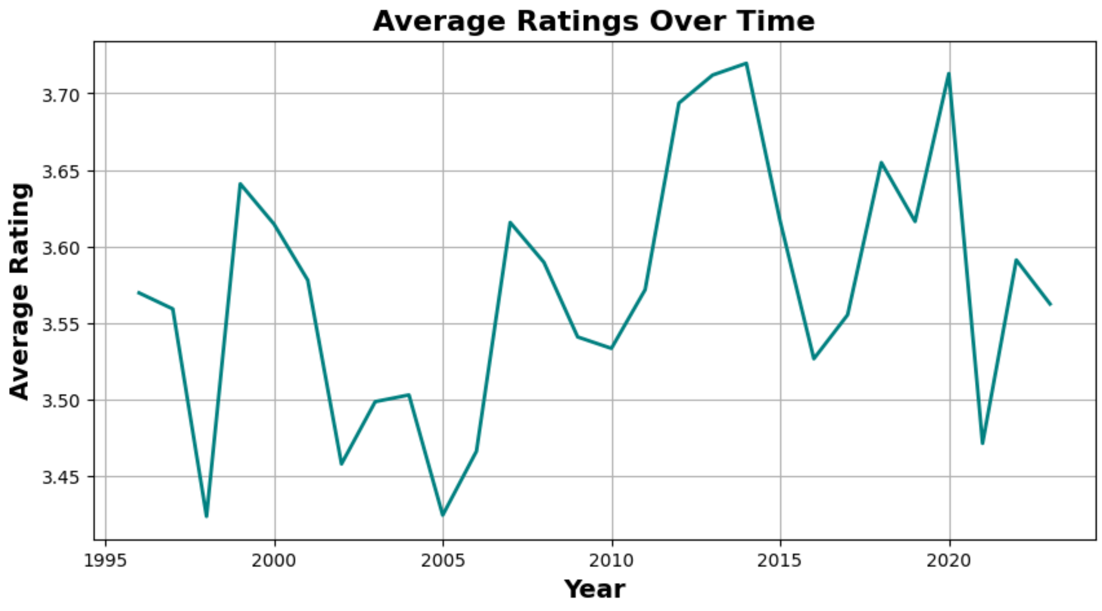


Figure 6.7: Average Rating Over Time

7. **Average Ratings Per User**

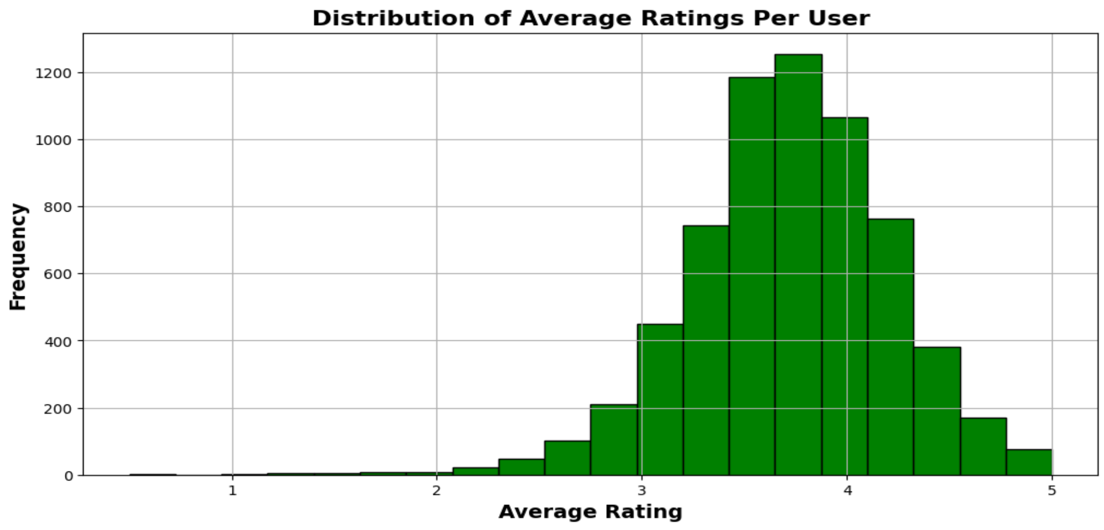


Figure 6.8: Average Rating Per User

The distribution of average ratings per user was visualized using a histogram. This plot provides insights into the rating behavior of individual users, showing how many users tend to give high, low, or average ratings.Most users have given an average rating in the range of 3-5.

8. **Top 10 Genres by Rating Count**

   This section focuses on the relationship between the number of movies per genre and the number of ratings. It determines the top 10 genres by the total number of ratings they receive. This helps to discover the most rated genres and how they differ in terms of total ratings.

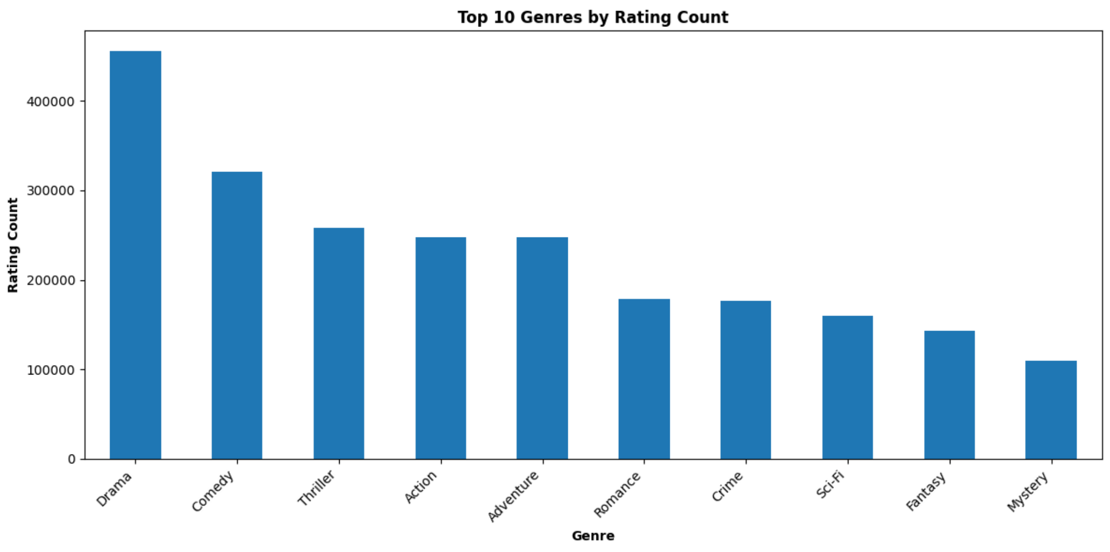


Figure 6.9: Top 10 Genres by Rating Count

6.2 Model Training Phase

The Training phase involves initial model training, where the selected models undergo training using the processed data. Each model learns patterns to make personalized recommendations.

6.2.1 Loss Graphs:

Loss graphs indicate how well a model is learning during training. A decreasing loss suggests the model is improving, while a fluctuating or stagnant loss may indicate issues like underfitting or overfitting. The loss graph of each model are:-

- **Content Based Training and Validation Loss:** The loss graph shows both training and validation loss curves over 20 epochs of a content-based model. Initially, both types of losses have a high value, validation loss being slightly higher than training loss. As the training goes on, both losses show a

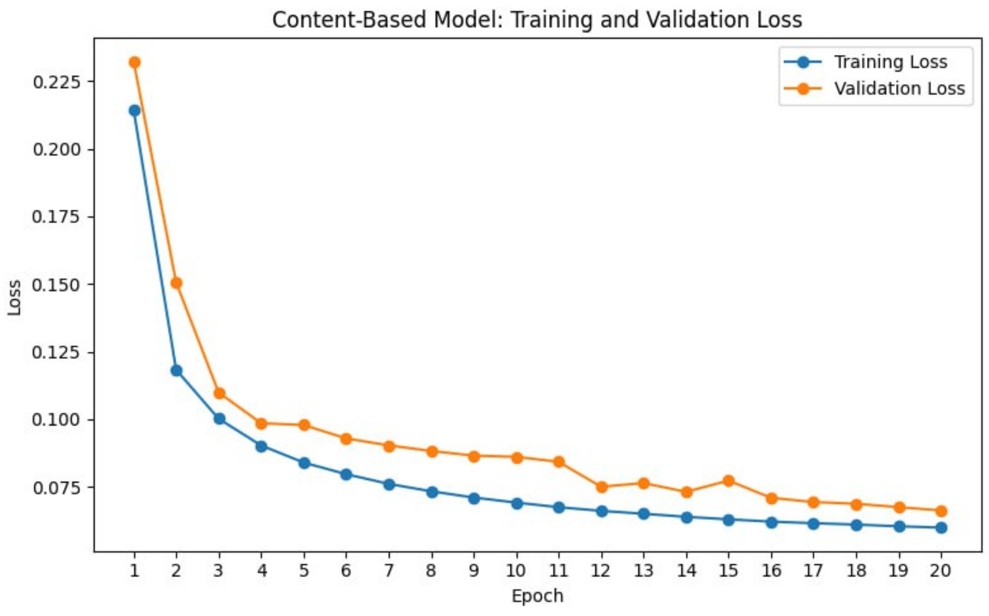


Figure 6.10: Graph showing Content-Based Training and Validation Loss

rapid decreasing trend in initial epochs and then gradually tend to stabilize, which is suggestive for efficient learning. The training and validation loss gap remains small, implying that there is less overfitting. At the last epoch (epoch 20), the training loss would be as 0.065 and validation loss would stand as 0.075, indicating a fairly low stable loss while generalization gap is still there in slight amount.

- **NeuMF Model Training and Validation Loss:**

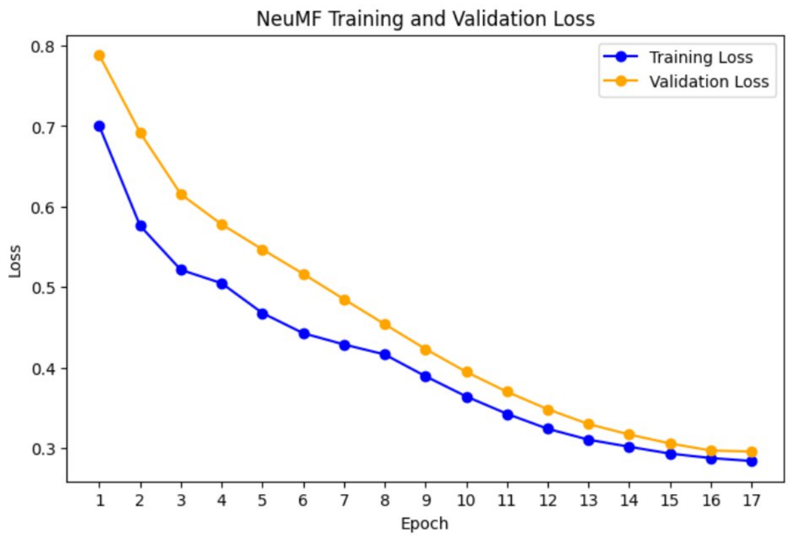


Figure 6.11: Graph showing NeuMF Model Training and Validation Loss

Basically, the loss graph shows the performance of the NeuMF model during 17 epochs of training, drawing out both training and validation losses. At the start of training, the losses were relatively high and have shown an impeccable downward trend, signifying a good learning process for the model-in other words, where data is concerned, the model seems to have appeared successful in reducing error. The training loss graph commences out at around 0.7 and steadily declines to 0.2817 epoch 17. The validation loss graph slightly above at 0.8 starts and settles to 0.2842 by the end of epoch 17. This convergence suggests that generalization is equally good concerning unseen data and fitting the training data as indicated by the closeness of training and validation losses at the end epoches.

- **GNN Model Training and Validation Loss:** The graph represents the

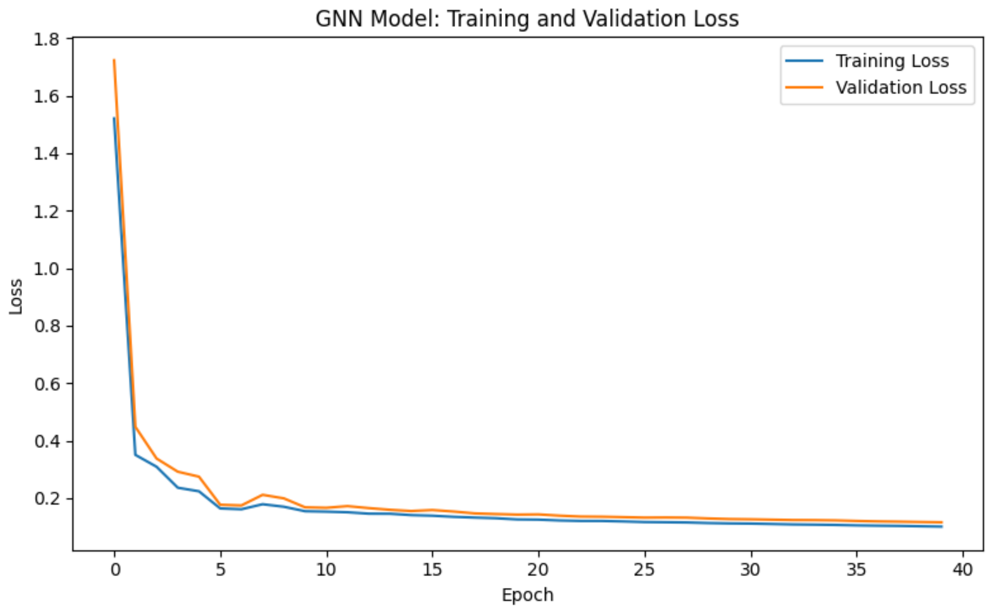


Figure 6.12: Graph showing GNN Model Training and Validation Loss

training and validation loss of the GNN model over 40 epochs. Both losses drop sharply in the initial epochs and gradually stabilize. The final training loss is 0.1010, while the validation loss is 0.1100, indicating good convergence and minimal overfitting.

6.3 Model Evaluation

The bar chart as well as the table present a comparison of three movie recommendation models: Content-Based, NeuMF, and GNN, utilizing five evaluation metrics. The lower the RMSE and MAE, the better the predictive accuracy of ratings; the higher the Precision@10, Recall@10, and NDCG@10, the stronger the recommendation quality. Hence, being the most accurate in predicting the ratings, NeuMF has the lowest RMSE (0.238) and MAE (0.189). Comparing to the benchmarks, GNN got much better in Precision@10 with the best value of 0.990 which states that it gives more relevant recommendations for the top 10 of those 10 item lists. Content Based lies lower in Recall@10: (0.180) while NeuMF occupies the top position in Recall (0.295) indicating that it retrieves more relevant items. GNN achieves the highest NDCG@10 (0.773), meaning its recommendations based on ranks are well-ordered. All in all, NeuMF is better at accuracy in rating aspects, while GNN is the best concerning ranking and precision.

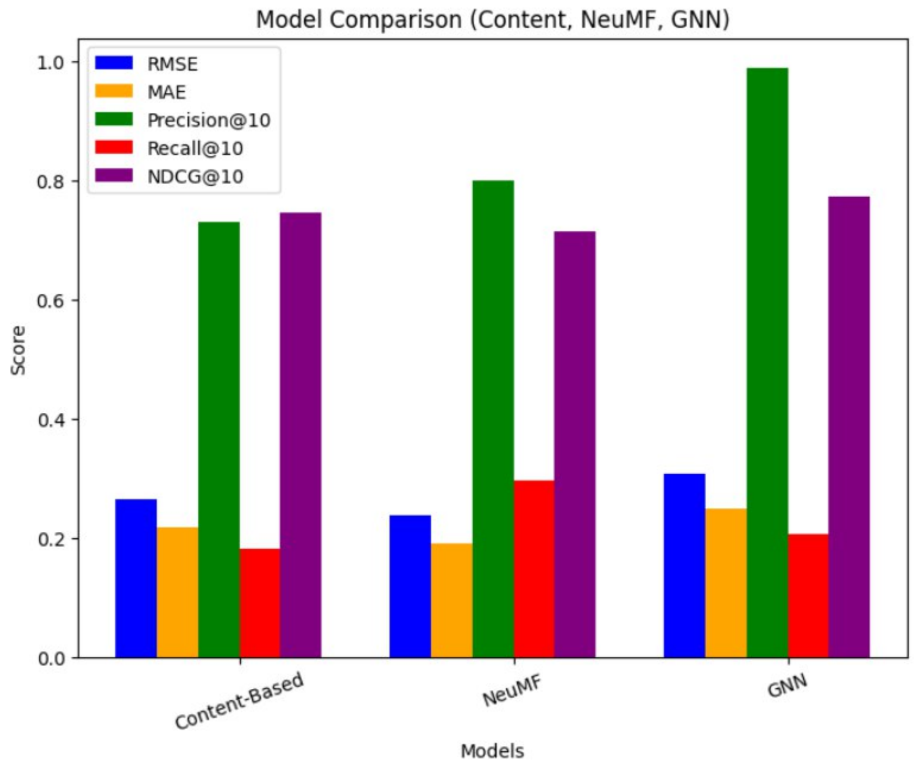


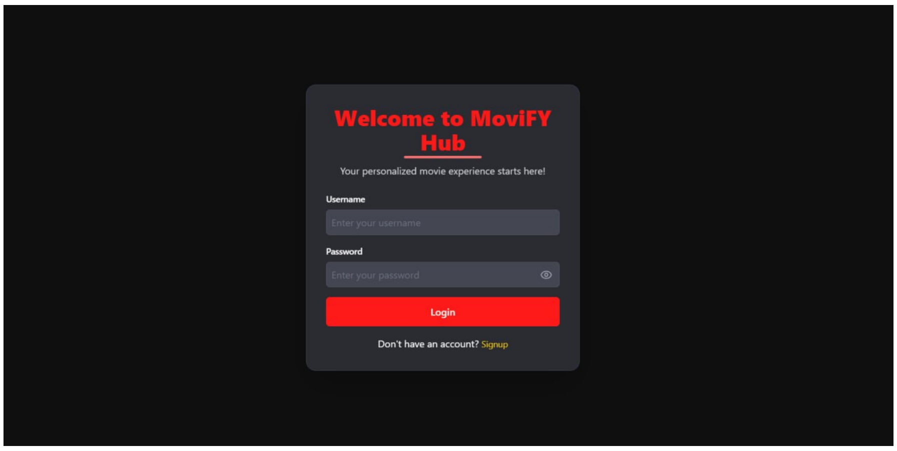
Figure 6.13: Performance Comparison of Models

The value of performance metrics are across NeuMF, GNN, and Content-Based Model:-

Table 6.1: Performance Comparison of Models

Models	RMSE	MAE	Precision@10	Recall@10	NDCG@10
Content-Based	0.265447	0.218332	0.73014	0.180945	0.745644
NeuMF	0.238369	0.189882	0.80000	0.295665	0.715521
GNN	0.307575	0.248725	0.99004	0.205261	0.773241

6.4 Outcomes


Welcome to MoviFY Hub

Your personalized movie experience starts here!

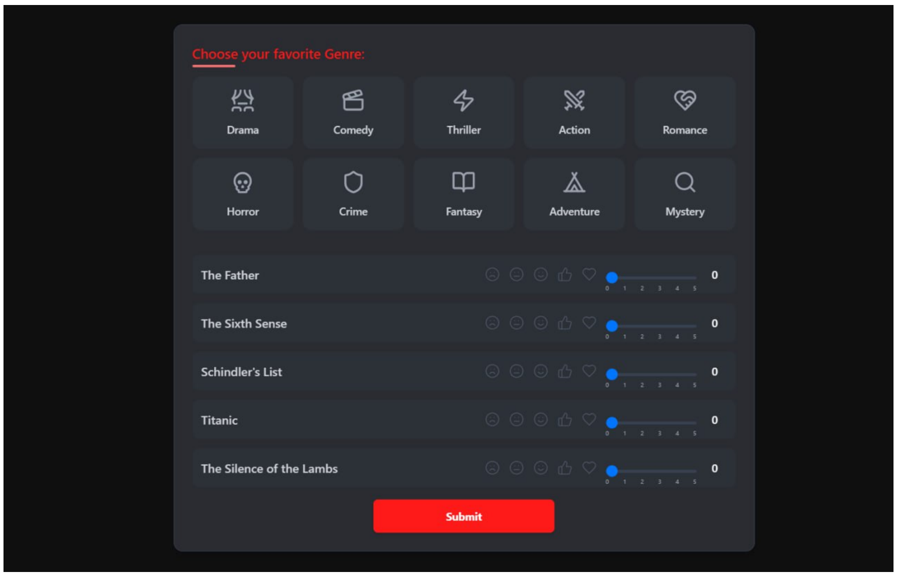
Username

Enter your username

Password

Enter your password

Figure 6.14: Login Page


Choose your favorite Genre:

The Father	0
The Sixth Sense	0
Schindler's List	0
Titanic	0
The Silence of the Lambs	0

Figure 6.15: Questionnaire

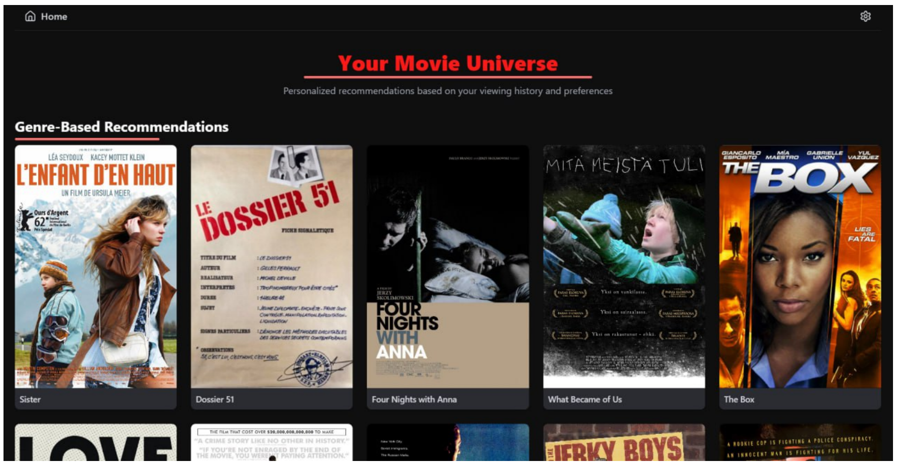

Home
Your Movie Universe
Personalized recommendations based on your viewing history and preferences
Genre-Based Recommendations
Sister
Dossier 51
Four Nights with Anna
What Became of Us
The Box

Figure 6.16: Recommendations

6.5 Limitations

This project has still limitations to address which are explained below:-

1. **Need Requisite Ratings:** The system performs optimally with sufficient ratings on several movies. In contrast, new users and those with few ratings track less accurate recommendations.

2. **Limited Descriptions:** The system accepts only genre and tagging information and has no input regarding anything else-audiences, directors, and movie styles.

3. **New Movies:** The recommendation system has no mechanism to accommodate completely new movies with no user ratings.

4. **Difficult Updates:** While the addition of new ratings is generally fast, the system might momentarily become quite inaccurate when a large volume of ratings comes flooding in.

6.6 Future Enhancements

Several future enhancements could address the limitations and further improve the system:

1. **Enhanced Cold Start:** Refine the way the system behaves for new users, possibly by firing better questions or using information from other websites.

2. **New Movie Handling:** Find alternatives to recommend new films, e.g. show it to some users, get an initial rating from them, and link it to one or more already rated films that have similarities with it.

3. **Intelligent Updating:** An improved approach for how the system updates itself when the new ratings come in when taking care that it doesn't overreact to sudden changes.

4. **Integration of a Larger Database:** For example, a bigger database having a multi-users capability as that of server-based databases like PostgreSQL can be used to store and handle bigger sets of information and secure better.

5. **Enhanced Security:** The protection of the system could be fortified through several other security mechanisms such as encryption to increase user privacy.

CHAPTER 7

Conclusion and Recommendation

This project has shown that a hybrid movie recommendation system can be developed that is effective in content-based filtering and neural collaborative filtering such as NeuMF and GNNs. The implemented system itself is a complete, operational web application with user authentication, personal recommendation, movie ratings, and a start mechanism for newcomers. The data and API management are handled well by the Django backend, while the user experience is optimized by the React frontend. Incremental model update integration allows the system to be well adapted to incoming user data. The weighted ensemble approach well for an accurate and adaptive system since they merge strengths of individual recommendation models. The system thus becomes personalized experience.
Further research should thus focus on enhancing personalized and engaging features for the user. For instance, improving cold-start techniques may arguably assist in attaining better initial recommendations by using more advanced active learning for preference elicitation strategies on new users. Adding explanations to the recommendations by showing the important features or similar users might further get such users engaged and more trusting of the system's recommendations. An investigation into the temporal dynamics in user preferences using more sophisticated GNN architectures may hold the key to making even better and more relevant recommendations over time. Finally, to complement the overall user experience, improving the user interface will allow for greater elements of interaction; for instance, allowing users to directly change the diversity of recommendations themselves. The introduction of these features would result in a better system overall.

REFERENCES

[1] Ching-Seh (Mike) Wu, Deepti Garg, and Unnathi Bhandary. Movie recommendation system using collaborative filtering, 2019.

[2] Yongheng Mu and Yun Wu. Multimodal movie recommendation system using deep learning. *Mathematics*, 11(4):895, 2023.

[3] Enliang Wang and Zhixin Sun. Research on personalized course resource recommendation method based on gemrec. *Applied Sciences*, 15(3):1075, 2025.

[4] Kai Li, Long Xu, Cheng Zhu, and Kunlun Zhang. A multimodal graph recommendation method based on cross-attention fusion. *Mathematics*, 12(15):2353, 2024.

[5] Sudhanshu Kumar, Shirsendu Sukanta Halder, Kanjar De, and Partha Pratim Roy. Movie recommendation system using sentiment analysis from microblogging data, 2018.

[6] Jiang Zhang, Yufeng Wang, Zhiyuan Yuan, and Qun Jin. Personalized real-time movie recommendation system: Practical prototype and evaluation. *Tsinghua Science & Technology*, 25(2):180–191, 2020.

[7] SRS Reddy, Sravani Nalluri, Subramanyam Kunisetti, S. Ashok, and B. Venkatesh. Content-based movie recommendation system using genre correlation. In *Smart Intelligent Computing and Applications, Proceedings of the Second International Conference on SCI 2018*, volume 2, pages 391–397, 2019.

[8] Guozhen Sang and Qin Xie. Construction of personalized movie recommendation model relying on recurrent neural network. *Journal of Electrical Systems*, 19(4), 2023.

[9] S. Sridhar, D. Dhanasekaran, and G. Charlyn Pushpa Latha. Content-based movie recommendation system using mbo with dbn. *Intelligent Automation & Soft Computing*, 35(3), 2023.

[10] Manoj Kumar, D.K. Yadav, Ankur Singh, and Vijay Kr. Gupta. A movie recommender system: Movrec. *International Journal of Computer Applications*, 124(3):7–11, 2015.

[11] Asha K. N. and R. Raj Kumar. Expectation maximization and ga-based movie recommender system. *EAI Endorsed Transactions on Scalable Information Systems*, 22(2):e2, July 2022.

[12] M. Sandeep Kumar and Prabhu J. Hybrid model for movie recommendation system using fireflies and fuzzy c-means. *International Journal of Web Portals*, 11(2):1–13, 2019.

[13] S. Rajarajeswari, Sharat Naik, Shagun Srikant, M. K. Sai Prakash, and Prarthana Uday. Movie recommendation system. In *Energy Transfer and Dissipation in Plasma Turbulence, Advances in Intelligent Systems and Computing*, pages 329–340, 2019.

[14] Ramni Harbir Singh, Sargam Maurya, Tanisha Tripathi, and Tushar Narula. Movie recommendation system using cosine similarity and knn. *International Journal of Engineering and Advanced Technology*, 9(5):2249–8958, 2020.

[15] Urvish Thakker, Ruhi Patel, and Manan Shah. A comprehensive analysis on movie recommendation system employing collaborative filtering. *Multimedia Tools and Applications*, 80(19), 2021.

[16] Triyanna Widiyaningtyas, Indriana Hidayah, and Teguh B. Adji. User profile correlation-based similarity (upcsim) algorithm in movie recommendation system. *Journal of Big Data*, 8:52, 2021.

[17] Hui Li, Jiangtao Cui, Bingqing Shen, and Jianfeng Ma. An intelligent movie recommendation system through group-level sentiment analysis in microblogs. *Neurocomputing*, 210, 2016.

Appendix

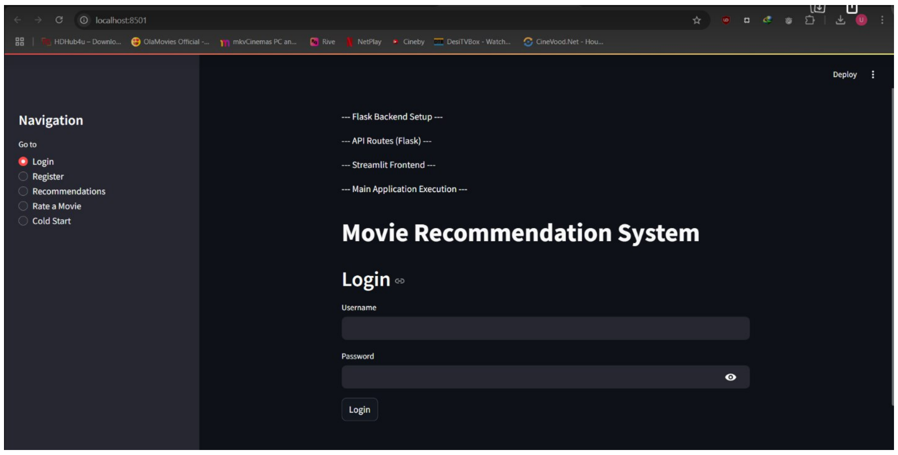
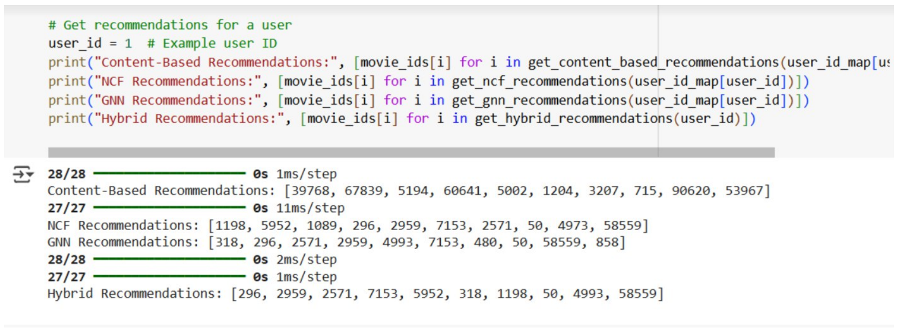

localhost:8501
Deploy
Navigation
Go to
Login
Register
Recommendations
Rate a Movie
Cold Start
--- Flask Backend Setup ---
--- API Routes (Flask) ---
--- Streamlit Frontend ---
--- Main Application Execution ---
Movie Recommendation System
Login
Username
Password
Login

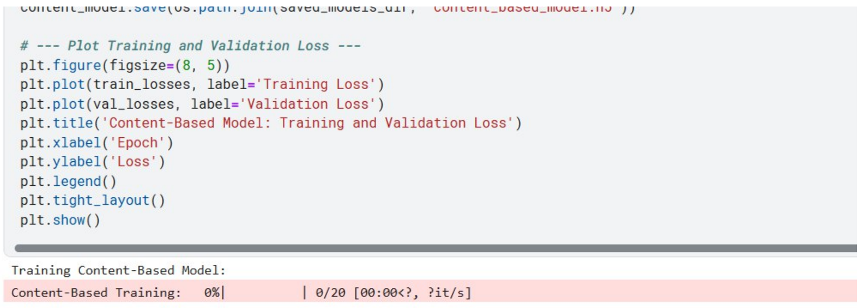
Figure A-1: Streamlit Implementation for System Testing



```
# Get recommendations for a user
user_id = 1  # Example user ID
print("Content-Based Recommendations:", [movie_ids[i] for i in get_content_based_recommendations(user_id_map[us
print("NCF Recommendations:", [movie_ids[i] for i in get_ncf_recommendations(user_id_map[user_id])])
print("GNN Recommendations:", [movie_ids[i] for i in get_gnn_recommendations(user_id_map[user_id])])
print("Hybrid Recommendations:", [movie_ids[i] for i in get_hybrid_recommendations(user_id)])
```

```
28/28 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
Content-Based Recommendations: [39768, 67839, 5194, 60641, 5002, 1204, 3207, 715, 90620, 53967]
27/27 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step
NCF Recommendations: [1198, 5952, 1089, 296, 2959, 7153, 2571, 50, 4973, 58559]
GNN Recommendations: [318, 296, 2571, 2959, 4993, 7153, 480, 50, 58559, 858]
28/28 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
27/27 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
Hybrid Recommendations: [296, 2959, 2571, 7153, 5952, 318, 1198, 50, 4993, 58559]
```

Figure A-2: Recommendation Testing

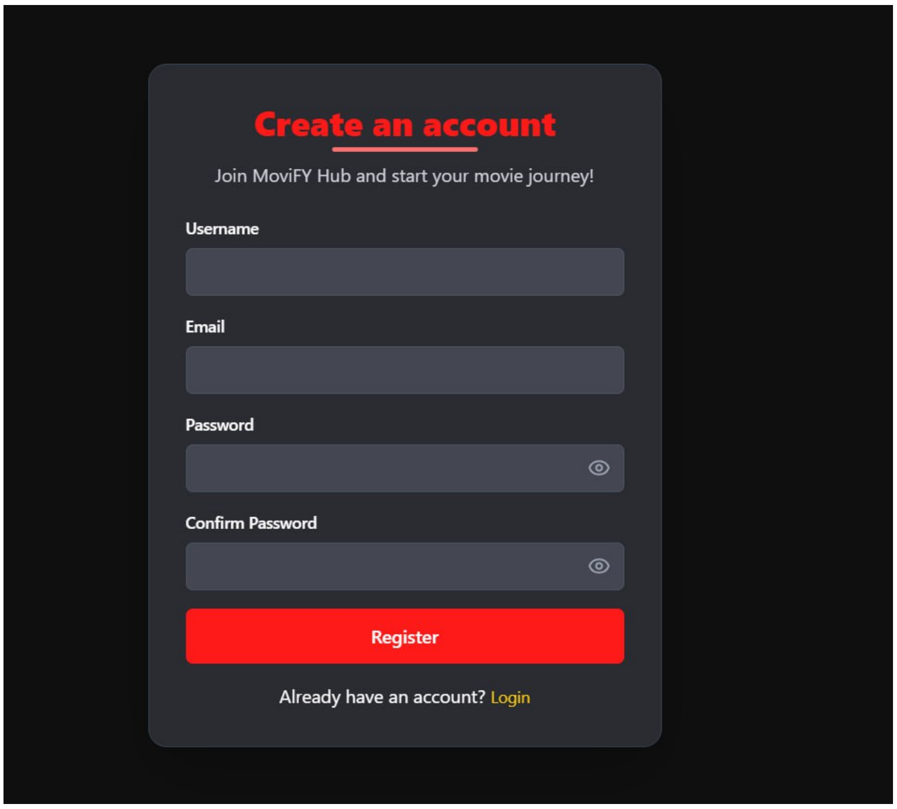


```
content_model.save(os.path.join(saved_models_dir, 'content_based_model.h5'))

# --- Plot Training and Validation Loss ---
plt.figure(figsize=(8, 5))
plt.plot(train_losses, label='Training Loss')
plt.plot(val_losses, label='Validation Loss')
plt.title('Content-Based Model: Training and Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.tight_layout()
plt.show()
```

```
Training Content-Based Model:
Content-Based Training:   0%|          | 0/20 [00:00<?, ?it/s]
```

Figure A-3: Model Training Progress in Kaggle



Create an account

Join MoviFY Hub and start your movie journey!

Username

Email

Password

Confirm Password

Register

Already have an account? Login

Figure A-4: Registration Page

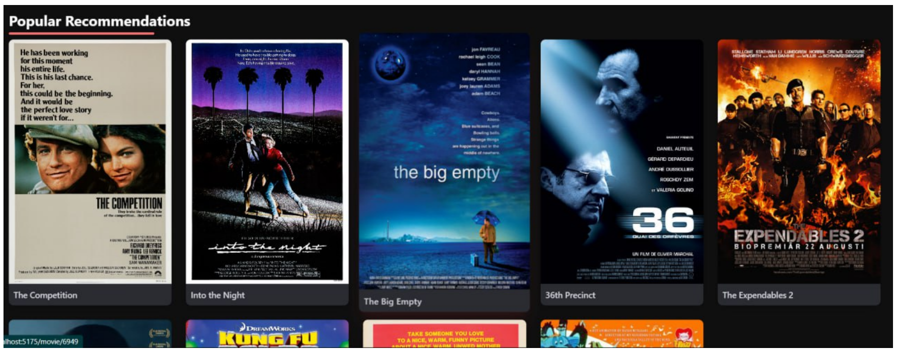
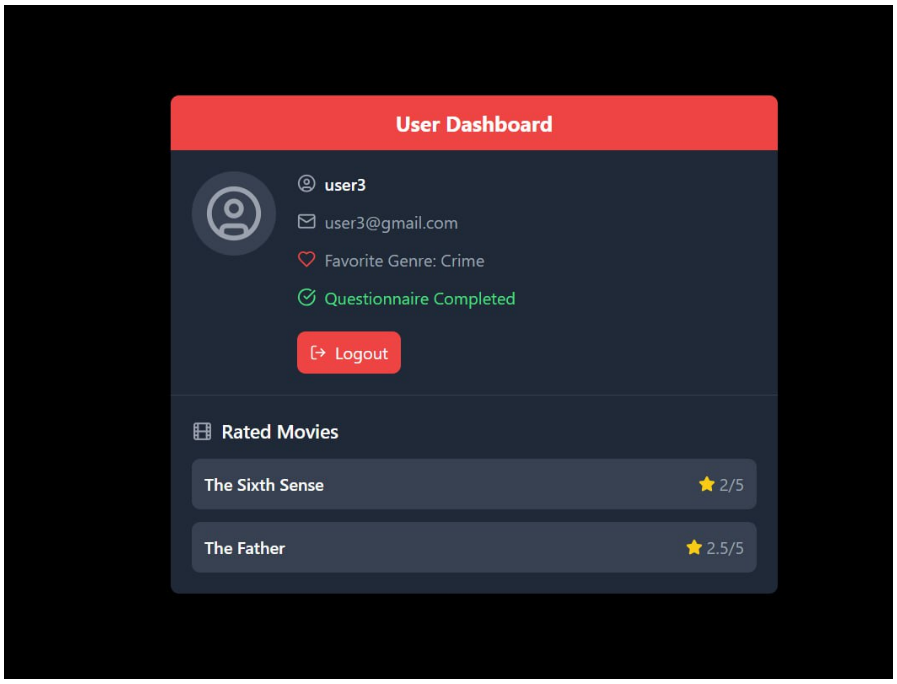


Figure A-5: Sample Recommendations Testing


User Dashboard

user3

user3@gmail.com

Favorite Genre: Crime

Questionnaire Completed

Rated Movies

The Sixth Sense	2/5
The Father	2.5/5

Figure A-6: User Dashboard