

Movie Recommendation System with Neural Network

Project Overview

This project implements a content-based movie recommendation system using a neural network. It leverages user-specific genre preferences and movie features (genres, year, average rating) to predict ratings for unseen movies, thereby generating personalized recommendations for a new user.

Features

- **Data Loading & Exploration:** Reads and explores `movies.csv` and `ratings.csv` datasets.
 - **Feature Engineering:**
 - Extracts movie genres and converts them into one-hot encoded features.
 - Extracts movie release year.
 - Calculates the average rating for each movie.
 - Calculates user genre preferences based on their historical ratings.
 - **Data Preprocessing:** Scales user and movie features using `StandardScaler` and `MinMaxScaler` for target ratings.
 - **Neural Network Model:** Implements a shallow neural network (user and movie embedding networks) to learn latent factors and predict movie ratings.
 - **Personalized Recommendations:** Generates a list of top 10 recommended movies for a new user based on their specified genre preferences.
-

Dataset

The system utilizes two primary datasets:

- **movies.csv**: Contains movie IDs, titles, and genres.
- **ratings.csv**: Contains user ratings for movies, including user ID, movie ID, rating, and timestamp.

These datasets are assumed to be present in the working directory.

Setup and Usage

Running the Notebook

1. **Environment**: This notebook is designed to be run in Google Colab.
2. **Dependencies**: Ensure you have the following libraries installed. They are typically pre-installed in Colab:
 - **pandas**
 - **numpy**
 - **tensorflow**
 - **scikit-learn**
3. **Data Files**: Upload `movies.csv` and `ratings.csv` to your Colab environment or ensure they are accessible from the notebook's working directory.
4. **Execute Cells**: Run all the code cells sequentially. The notebook performs data loading, preprocessing, model training, and generates recommendations.

Model Architecture

The recommendation system uses a two-tower neural network architecture:

- **User Network**: Processes user genre preferences and average rating through Dense layers (128, 64 neurons) to generate a user embedding (32 dimensions).

- **Movie Network:** Processes movie features (year, average rating, genres) through Dense layers (128, 64 neurons) to generate a movie embedding (32 dimensions).
 - **Prediction:** The dot product of the normalized user and movie embeddings predicts the rating. The model is compiled with Adam optimizer and MeanSquaredError loss.
-

Generating Recommendations for a New User

To get recommendations for a new user, modify the new_user_dict in the relevant code cell to reflect their preferences across different genres. For example:

```
new_user_dict = {  
    'Adventure' : 0,  
    'Animation' : 0,  
    'Children' : 0,  
    'Comedy' : 0,  
    'Fantasy' : 0,  
    'Romance' : 0,  
    'Drama' : 0,  
    'Action' : 0,  
    'Crime': 0,  
    'Thriller' : 4, # User likes Thriller with a preference score of 4  
    'Horror' : 0,  
    'Mystery' : 0,  
    'Sci-Fi' : 5, # User highly likes Sci-Fi with a preference score of 5  
    'War' : 0,  
    'Musical' : 0,  
    'Documentary' : 0,  
    'IMAX' : 0,  
}
```

```

    'Western' : 0,
    'Film-Noir': 0,
    '(no genres listed)': 0,
}

# The 'avg_rating' is automatically calculated based on non-zero preferences.

```

After updating the new_user_dict, re-run the cells that process new user features and generate recommendations.

Example Results

Below is an example of top 10 movie recommendations for a user who shows high preference for 'Thriller' and 'Sci-Fi' genres:

Title	Predicted Rating
Assignment, The (1997)	4.807502
Knock Off (1998)	4.794699
Alien Contamination (1980)	4.787239
Nirvana (1997)	4.779356
Maniac Cop 2 (1990)	4.774135
Supercop 2 (Project S) (Chao ji ji hua) (1993)	4.763146
Dog Soldiers (2002)	4.741652
Matrix, The (1999)	4.735595
Galaxy of Terror (Quest) (1981)	4.726744
Branded to Kill (Koroshi no rakuin) (1967)	4.723955

(Note: These specific recommendations might vary slightly based on model re-training or subtle changes in inputs.)

Further Improvements

- **Hybrid Approach:** Integrate collaborative filtering techniques for a more robust system.
- **Temporal Dynamics:** Incorporate time-based features to account for evolving user preferences and movie trends.
- **Cold Start Problem:** Implement strategies to handle new users or new movies with limited data.
- **Deep Learning Architectures:** Experiment with more complex neural network models like Recurrent Neural Networks (RNNs) for sequential data or Transformer models.
- **Explainability:** Add features to explain *why* a particular movie was recommended.