Deep Learning Course Project
Final Report

Problem Statement: Design a deep learning model for a Personalised Movie Recommendation System.

Introduction:
Developing a personalised movie recommendation system using deep learning techniques poses a significant challenge due to the complex nature of user preferences and the vast amount of available movie data. The objective of this project is to develop a deep learning model for a Personalised Movie Recommendation System. The system will utilise the capabilities of deep learning algorithms to examine user preferences and past experiences with films in order to produce tailored suggestions.

Objectives:
- Developing a deep learning architecture capable of effectively capturing and understanding user preferences.
- Training the model on a diverse dataset of movie ratings and user interactions.
- Evaluating the performance of the recommendation system based on relevant metrics

Dataset:
For our project, we are utilising the MovieLens dataset, which is a widely used benchmark dataset for recommendation systems research. The MovieLens dataset contains user ratings and movie metadata, including genres, tags, and titles, collected from the MovieLens website. The dataset is available in various sizes, ranging from 100,000 ratings to 27 million ratings, allowing for scalability and flexibility in experimentation.

Dataset	Parameters	Value
MovieLens 100 K	#Users	943
	#Movies	1682
	#Ratings	100,000
	Sparsity	93.695%
MovieLens 1 M	#Users	6040
	#Items	3883
	#Ratings	1,000,209
	Sparsity	95.359%

The MovieLens dataset provides a rich source of information for training and evaluating recommendation models. It includes user-item interactions, which serve as the basis for learning user preferences and item features.
Dataset link: Movielens Dataset

Solution strategy :

The strategy is to provide personalised and relevant recommendations to users by combining the strengths of content-based and user-based recommendation approaches.

Content-Based Recommendation:

- In a content-based recommendation system, recommendations are made based on the attributes or features of the items themselves.
- This approach focuses on understanding the characteristics of items (e.g., movies, genres) and recommending similar items to users based on their preferences.
- The system analyses the content or metadata of items, such as genres, to build a profile of each item.
- Content-based recommendation excels at recommending items that are similar in content or theme to those previously rated or interacted with by users.

Implementation of Content-Based Recommendation:

- We designed and trained a neural network model to predict ratings for items (movies) based on features extracted from their content or metadata.
- The input features consisted of high-dimensional representations of movie genres, in the form of one-hot encoded vectors. To optimise this, an Autoencoder is trained with the objective of learning a compressed and meaningful representation of the movie genres in a lower-dimensional latent space.
- The new features are trained with the MLP model and ratings of the movies which are not rated by the user can be predicted.
- Before giving the unrated movies to the model, filtering is done as realistically the model should only give predictions for the movies containing the genres which have been rated by the user. Filtering gives the unrated movies which contain at least one genre from the movies rated by the user.
- Predictions are done and sorted by ratings and top N movies with higher ratings are given as recommendations.

User-Based Recommendation:

- In a user-based recommendation system, recommendations are made based on the preferences or behaviour of similar users.
- This approach focuses on identifying users with similar tastes or preferences and recommending items that those users have rated or interacted with.
- Similarity metrics such as cosine similarity are used to measure the similarity between users' profiles.
- Recommendations are then generated by identifying items that similar users have liked but the target user has not yet interacted with.
- User-based recommendation systems are effective for scenarios where users preferences are diverse and where leveraging the wisdom of the crowd can improve recommendation quality.

Implementation of User-based Recommendation:

- We designed and trained complex networks like RNN,LSTM and GNN to predict the top recommended movies for a given user id based on the rating given by similar users.
- The **RNN** model takes as input user and item IDs, which are embedded into dense representations using embedding layers where the embeddings are concatenated and passed through an RNN layer, the model learns temporal dependencies between user-item interactions
- The RNN output is passed through a fully connected layer (linear transformation) to predict a single output, which represents the recommendation score for the given user-item pair.
- The **LSTM** model takes sequences of user data(ratings) as input. The LSTM layer processes this data, capturing temporal dependencies and patterns in the user's preferences over time.
- The LSTM outputs a sequence of hidden states, representing the model's understanding of the user's preferences at each time step. This sequence is a learned representation of the user's taste in movies.
- The final hidden state from the LSTM is passed through a fully connected layer to generate the output, which represents the model's predictions for recommended movies that align with the user's interests.
- The **GNN** model takes user and item IDs as input, which are embedded into dense representations using embedding layers. These embeddings are then fed into a

GNN layer, specifically a Gated Recurrent Unit (GRU) cell, which iteratively updates the hidden state representation of the user embeddings. The GNN layer is iterated over a fixed number of steps, allowing the model to capture relational dependencies between users and items.

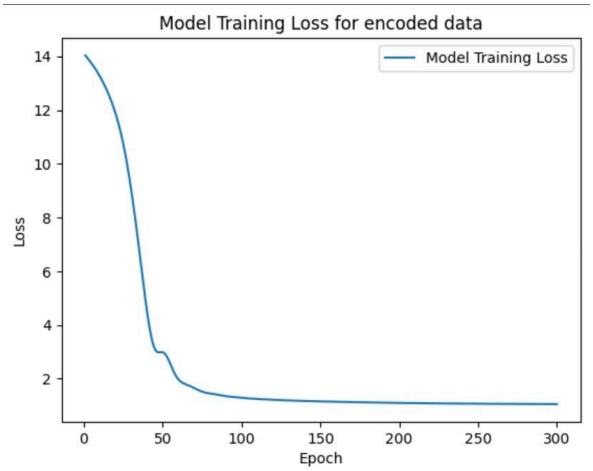
- The GNN layer, the user embeddings are concatenated with the item embeddings, and passed through a fully connected layer to produce a single output, representing the recommendation score for the user-item pair.
- The **GraphSAGE** model implements a movie recommendation system using a Graph Neural Network (GNN) approach with SAGEConv layers from the DGL library.
- The GraphSAGE model starts by loading movie ratings data and creating a graph with users and movies as nodes, assigning zero vector features to movie nodes. It uses a MovieRecommender architecture with two SAGEConv layers for message passing and aggregation.
- Recommendations are made for users based on movies they have not yet rated. For each user, embeddings are obtained for all movies, and cosine similarity is computed between the user's rated movies and all other movies to recommend those with the highest similarity scores

Innovations :

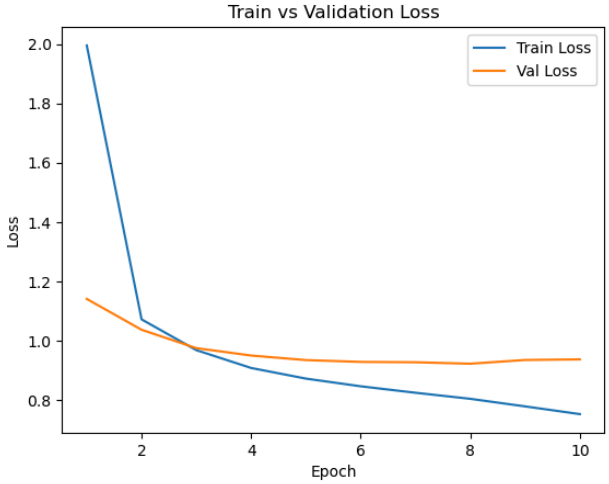
- Integration of both content-based and user-based recommendation approaches, leveraging the strengths of each method to enhance recommendation accuracy, diversity, and personalization.
- Exploration of sequential user interactions with movies using RNN and LSTM models to capture temporal dynamics in user preferences.
- Utilisation of graph representation and GNNs to model complex relationships between users and items in a collaborative filtering setting.

Results :

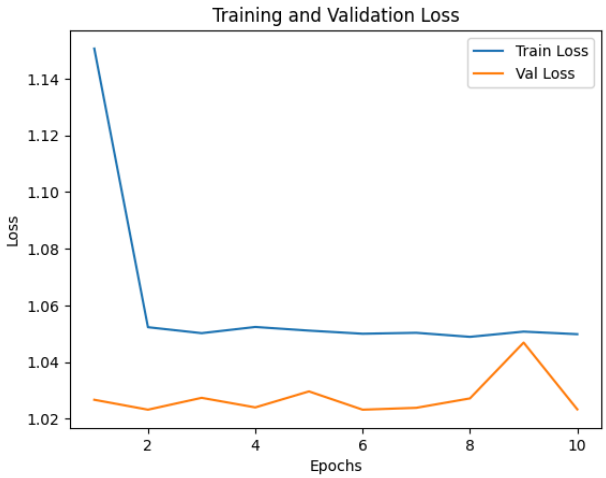
- The performance of each model was evaluated using metrics such as Mean Squared Error (MSE) and the losses for each epoch are plotted.



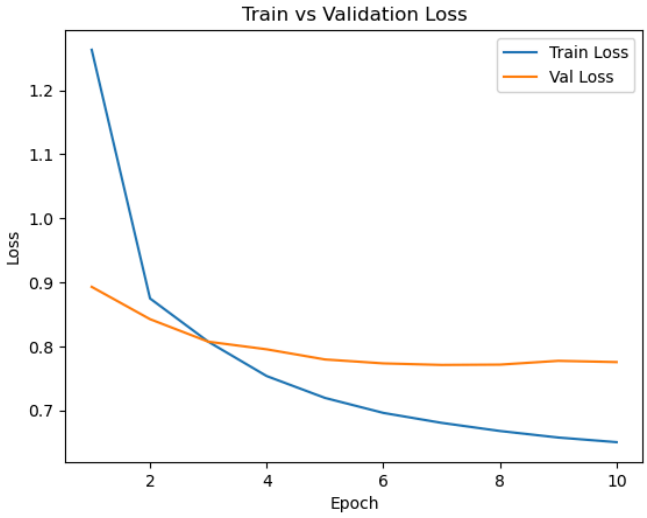
- The test loss calculated for autoencoder is **0.04574** and the test loss for the MLP model calculated is **1.04269**



- The test loss calculated for RNN is **0.9559**



- The test calculated for LSTM is **1.0238**



- The test loss calculated for GNN is **0.7754** and the mean squared error for the GraphSage model calculated is **7.6214**

Analysis of the solution with discussion on the possible weaknesses:

Strengths:

- ➢ Diverse set of models allows for comprehensive exploration of different recommendation strategies.
- ➢ Integration of content-based features enhances recommendation quality and addresses the cold start problem.
- ➢ Models such as RNN and LSTM capture sequential user interactions and temporal dynamics, improving recommendation accuracy.

Weaknesses:

- ➢ Models may suffer from scalability issues with larger datasets due to their complexity.
- ➢ Cold start problem for new users and items may still persist despite content-based features.
- ➢ Interpretability of deep learning models may be limited, making it challenging to understand why certain recommendations are made.
- ➢ It can be challenging to construct an adjacency matrix when dealing with a large number of features, especially in a content-based Graph Neural Network (GNN) context. This is because the model needs to accurately represent complex relationships between users and items based on various content features such as

genres, tags, and other attributes. A high number of features can lead to high-dimensional, sparse matrices that are difficult to process and require significant computational resources.

Conclusion:

In conclusion, the personalised movie recommendation system developed through this project offers a range of innovative approaches to address the challenges of traditional recommendation systems. By leveraging deep learning techniques and integrating content-based information, the system achieves improved recommendation quality and user satisfaction. However, further research is needed to address scalability issues and enhance the interpretability of models. Overall, our project contributes to the advancement of recommendation systems and provides valuable insights for future research in the field.

References:

- https://arxiv.org/abs/1706.02216
- https://www.turing.com/kb/collaborative-filtering-in-recommender-system
- https://medium.com/coderhack-com/building-a-recommendation-engine-with-pytorch-b2a2982ea2c9
- https://www.researchgate.net/publication/345244432_Long_Short-Term_Memory_Based_Movie_Recommendation
- https://www.dgl.ai/
- https://www.sciencedirect.com/science/article/pii/S0952197623007376?via%3Dihub

Codebase : *Colab.ipynb*

Team Members :

1. Gandyadapu Sriharsha (B21CS029)
2. Gavva Sathwika (B21CS030)
3. Dendi Sai Charitha (B21BB008)
4. Pappusani Sai Kiran Reddy (B21CS056)