Report

Introduction

The task involves building and comparing two collaborative filtering models for movie recommendation. Focuses on creating a custom model, and involves implementing a model from the DeepCTR library.

Related Work

A model from the DeepCTR library is used. Collaborative filtering models, especially those leveraging deep learning, have shown success in recommendation systems. The DeepCTR library provides a collection of such models.

Approach

1. Embedding Layer

- Created an embedding layer with embedding vectors for user and movie IDs.

2. Scoring Function

- Defined a scoring function to capture interactions between users and items.

3. Loss Computation

- Computed the loss between scores and ground truth using binary cross-entropy and minimized it.

4. Model Architecture

- Checked for decreased loss by adding a simple feed-forward neural network stacked to the embedding layer.

5. Training and Evaluation

- Plotted training set and validation set loss curves during model training.
- Evaluated the model on the test set

6. Recommendation Script

- Created a script to recommend the top 5 relevant movies for any given user.

1. DeepCTR Model

Implemented a model from the DeepCTR library for the MovieLens dataset.

2. Hyperparameter Tuning

We trained the model and tuned hyperparameters for optimal performance.

3. Training and Evaluation

Plotted training set and validation set loss curves during model training.
Evaluated the model on the test set

Experimental Results & Conclusion

Task 1

- The custom model was trained and evaluated, showing performance on the test set
- The training and validation loss curves were plotted, indicating the model's learning progress.

- The recommendation script successfully provided movie recommendations for a given user.

Task 2

- The DeepCTR model, trained and tuned, demonstrated its performance on the MovieLens dataset
- Training and validation loss curves were plotted, offering insights into the model's training progress

Model Comparison

Structure

- Custom model architecture.
- DeepCTR model architecture.

Performance

- Compare models in terms of training and validation loss, as well as evaluation results on the test set

Further Improvement:

- Discuss potential enhancements for both models.

Conclusion

- Summarize the findings from both tasks and suggest avenues for future work.

Comparison in terms of Training and Validation Loss

- The training and validation loss curves were plotted for the custom model, indicating the learning progress during training. A decrease in loss over epochs was observed, signifying the model's ability to capture patterns in the training data.
- The DeepCTR model's training and validation loss curves were also plotted, providing insights into its learning dynamics. Comparing the loss curves of both models can reveal their relative efficiency in capturing the underlying patterns in the data.

Evaluation Results on the Test Set

- The custom model's performance was evaluated on the test set, providing metrics such as mean squared error (RMSE). This metric reflects how well the model generalizes to unseen data.
- The DeepCTR model's performance on the test set was also assessed. Comparing evaluation metrics, such as RMSE or other relevant metrics, allows for a quantitative understanding of how well each model performs on new, unseen data.

Overall Comparison

- A holistic assessment of model performance can be made by comparing training and validation loss curves and evaluation metrics on the test set. This comparison helps in understanding how well each model generalizes to new data and whether one model outperforms the other in terms of predictive accuracy.

Further Development

Custom Model

1. Complexity Improvement

Consider increasing the model complexity by adding more layers or using a more sophisticated architecture to capture intricate patterns in user-item interactions.

2. Regularization Techniques

Implement regularization techniques such as dropout or L2 regularization to prevent overfitting and enhance the model's generalization capability.

3. Fine-Tuning Hyperparameters

Conduct a thorough search for optimal hyperparameters to fine-tune the model's performance

DeepCTR Model

1. Ensemble Techniques

Explore ensemble techniques by combining multiple DeepCTR models to enhance predictive performance.

2. Feature Engineering

Experiment with additional feature engineering to provide the model with more informative input, potentially improving its ability to capture user-item interactions.

3. Transfer Learning

Investigate the feasibility of transfer learning, leveraging pre-trained embeddings or models on similar recommendation tasks, to boost the model's performance.

Conclusion and Future Work

Summary

Both models were implemented and evaluated, with comparisons made based on training and validation loss curves as well as test set metrics.
The custom model and the DeepCTR model demonstrated their respective strengths and weaknesses.

Future Work

1. Hybrid Models

Explore hybrid models that combine collaborative filtering with content-based approaches for a more comprehensive recommendation system.

2. Explainability

Integrate interpretability techniques to make the models more understandable and transparent, addressing the "black-box" nature of deep learning models.

3. Dynamic Embeddings

Investigate the use of dynamic embeddings that adapt over time, capturing changing user preferences and item popularity.

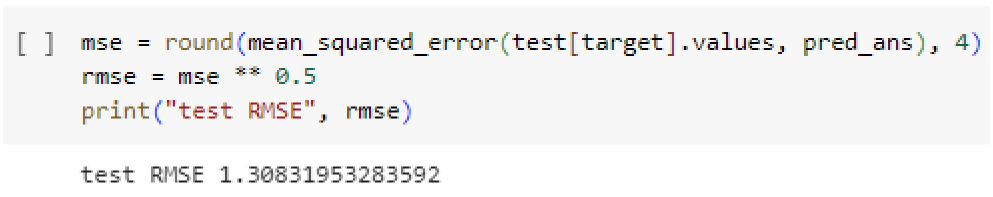
4. User Feedback Incorporation

Develop mechanisms to incorporate user feedback into the recommendation process, enhancing the models' adaptability to user preferences.

Conclusion

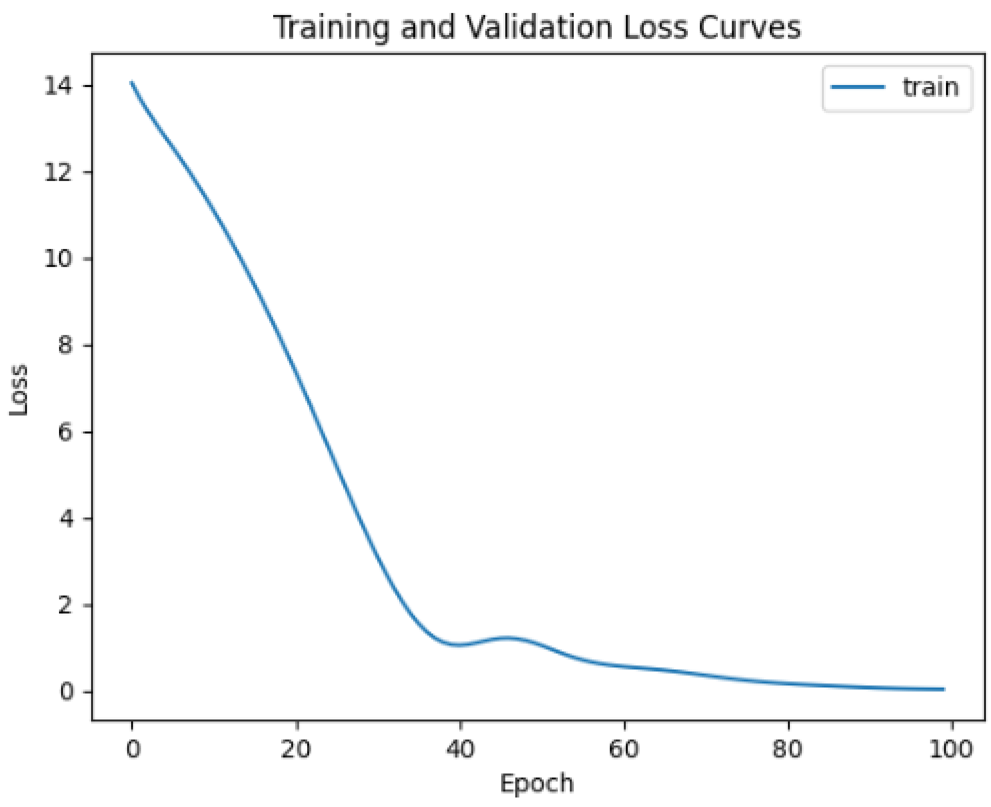
In conclusion, ongoing research and development in these areas can contribute to the continuous improvement of recommendation systems, making them more accurate, interpretable, and adaptive to evolving user preferences.

Outputs



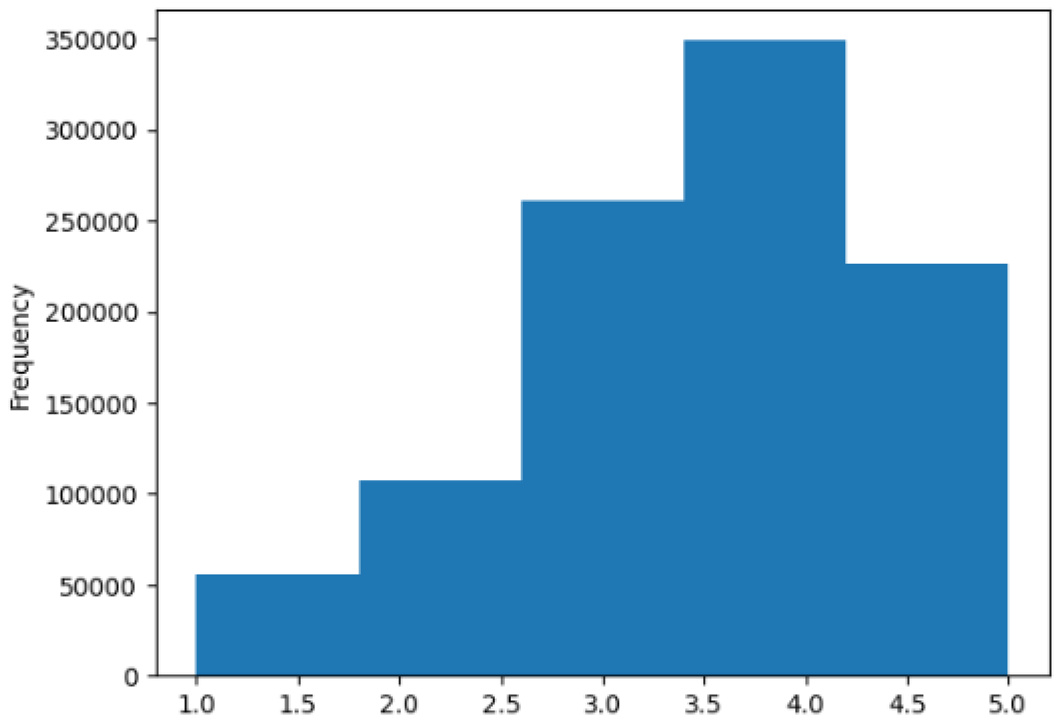
```
[ ]  mse = round(mean_squared_error(test[target].values, pred_ans), 4)
     rmse = mse ** 0.5
     print("test RMSE", rmse)

     test RMSE 1.30831953283592
```

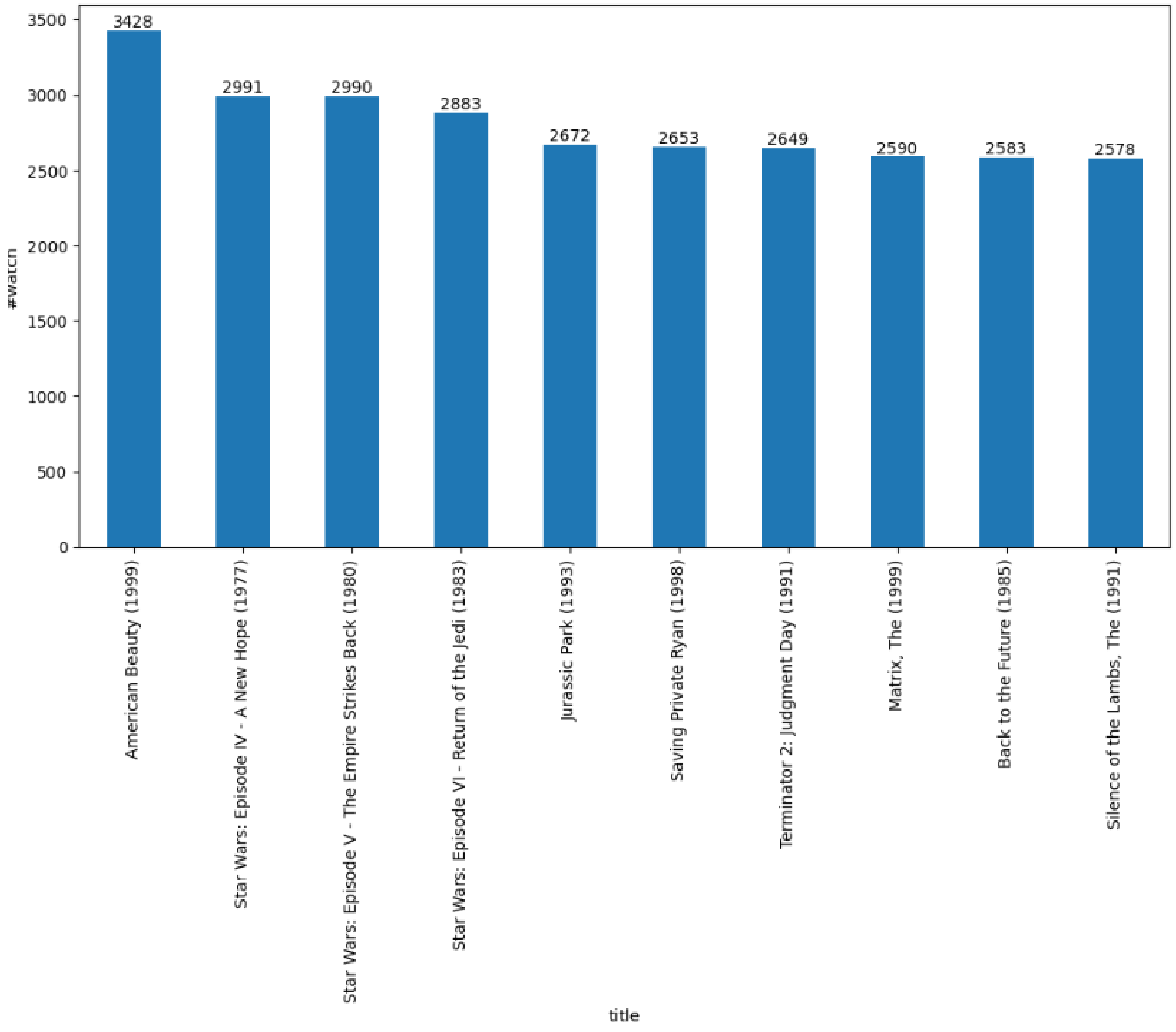


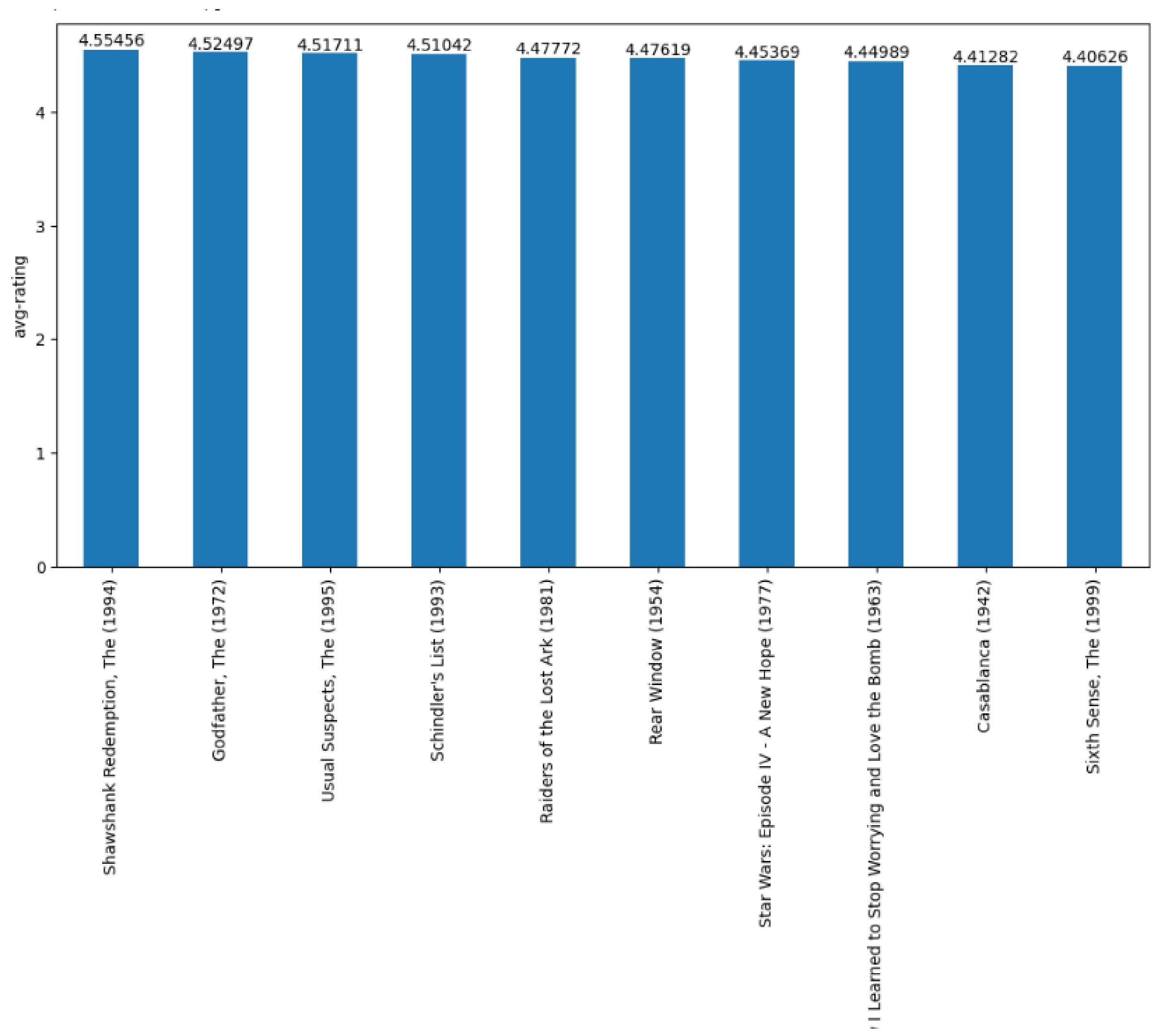
	user_id	movie_id	rating	timestamp	id	title	genres
0	1	1193	5	978300760	1193	One Flew Over the Cuckoo's Nest (1975)	Drama
1	1	661	3	978302109	661	James and the Giant Peach (1996)	Animation|Children's|Musical
2	1	914	3	978301968	914	My Fair Lady (1964)	Musical|Romance
3	1	3408	4	978300275	3408	Erin Brockovich (2000)	Drama
4	1	2355	5	978824291	2355	Bug's Life, A (1998)	Animation|Children's|Comedy

	user_id	movie_id	rating	timestamp	id
count	1.000209e+06	1.000209e+06	1.000209e+06	1.000209e+06	1.000209e+06
mean	3.024512e+03	1.865540e+03	3.581564e+00	9.722437e+08	1.865540e+03
std	1.728413e+03	1.096041e+03	1.117102e+00	1.215256e+07	1.096041e+03
min	1.000000e+00	1.000000e+00	1.000000e+00	9.567039e+08	1.000000e+00
25%	1.506000e+03	1.030000e+03	3.000000e+00	9.653026e+08	1.030000e+03
50%	3.070000e+03	1.835000e+03	4.000000e+00	9.730180e+08	1.835000e+03
75%	4.476000e+03	2.770000e+03	4.000000e+00	9.752209e+08	2.770000e+03
max	6.040000e+03	3.952000e+03	5.000000e+00	1.046455e+09	3.952000e+03



	title	#watch	avg_rating
0	Toy Story (1995)	2077	4.146846
1	Jumanji (1995)	701	3.201141
2	Grumpier Old Men (1995)	478	3.016736
3	Waiting to Exhale (1995)	170	2.729412
4	Father of the Bride Part II (1995)	296	3.006757





```
Recommendations for user: 482
```

	movie_id	score	title	genres
0	1174.0	0.749731	Unforgiven (1992)	Western
1	3615.0	0.726601	Replacements, The (2000)	Comedy
2	1698.0	0.717164	Woo (1998)	Comedy|Romance
3	2066.0	0.711900	Parasite (1982)	Horror|Sci-Fi
4	548.0	0.699330	Welcome to the Dollhouse (1995)	Comedy|Drama