Hybrid Recommender System

Bansil Patel
Artificial Intelligence and Machine Learning
Lambton College
Toronto, Canada
c0902873@mylambton.ca

Harsh Mohile
Artificial Intelligence and Machine Learning
Lambton College
Toronto, Canada
c0912872@mylambton.ca

Meet Patel
Artificial Intelligence and Machine Learning
Lambton College
Toronto, Canada
c0910378@mylambton.ca

Rachit Bhatt
Artificial Intelligence and Machine Learning
Lambton College
Toronto, Canada
c0902810@mylambton.ca

***Abstract*—This article describes a system that recommends using collaborative and content-based filtering for users predicting a movie. With the help of Matrix Factorization and neural networks, achieved hybrid recommendation system with EDA. Use of MovieLens dataset and tweets related to movies, comparing both techniques using RMSE, MAE, Precision, Recall, and F1-Score. The hybrid system implies a neural embedding layer in PyTorch, resulting in effective integration of content-based and collaborative filtering with sentiments, and highlighting improvements in accuracy as well as user satisfaction.**

Keywords—rotten tomatoes, comments, analysis, sentiment analysis, social media analysis, web scraping, user engagement, data visualization.

I. Introduction

The recommended systems are vital in modern data-driven applications, enabling personalized user experiences by suggesting items of interest. Traditional systems include content-based filtering, collaborative filtering, and hybrid methods [3]. The collaborative filtering estimates the user preferences based on their historical interactions, while the hybrid model integrates both approaches and enhances user experience [4]. With the help of Matrix Factorization in Collaborative Filtering and NN layers in PyTorch, the prediction of content for the user based on their preferences becomes realistic and the significance of their comparison expands the understanding the impact of incorporating diverse data sources from Rotton Tomatoes [5].

II. Related Study

Recommender systems have been extensively studied across various domains, and different techniques have been proposed to enhance their performance. He et al. [1] introduced neural collaborative filtering, demonstrating its efficiency in capturing user-item interactions with neural networks. Koren et al. [2] presented a foundational technique for collaborative filtering, leveraging latent factor models for accurate recommendations. Item-to-item collaborative filtering used by Amazon, as presented by Linden et al. [3], excels in scalability and precision.

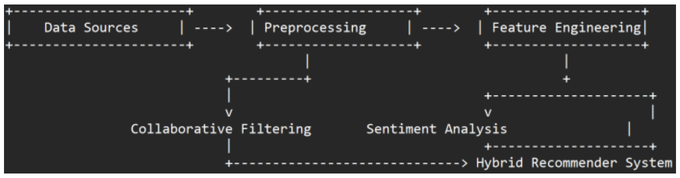
Comparative studies such as Said and Mauro's [4] underscore the strengths and weaknesses of different collaborative filtering algorithms. Ricci et al. [5] provided a comprehensive overview of recommender systems, emphasizing hybrid approaches. The implementation of matrix factorization that sets a benchmark for the recommendation accuracy as documented by Funk [6] behind Netflix's success. Sarwar et al. [7] introduced item-based algorithms, proving their robustness in sparse datasets.

The trust-aware systems integrate social trust into recommendations and enhancing user satisfaction as explored by Jamali and Ester [8]. The survey of Burke [9] on hybrid systems demonstrated their versatility in combining collaborative and content-based methods. Aggarwal [10] emphasized content-based recommenders' adaptability in niche domains, paving the way for advanced hybrid models. These studies collectively establish a foundation for this project's exploration of collaborative filtering, content-based filtering, and a hybrid model.

III. Methodology

This component outlines the methodologies for building collaborative filtering and hybrid recommender systems. We use the Matrix Factorization algorithm for collaborative filtering and PyTorch's embedding layer for the hybrid model [2][5].

A. System Architecture



```
+----------------------+      +-------------------+       +-------------------+
|    Data Sources     | ---->  | Preprocessing    | ---->  | Feature Engineering|
+----------------------+      +-------------------+       +-------------------+
                                     |                              |
                            +--------+                              +
                            |                              +--------------------+
                            v                              v                    |
                Collaborative Filtering      Sentiment Analysis              |
                            |                              +--------------------+
                            +----------------------------> Hybrid Recommender System
```

Fig. 1. Process Flow for Sentiment Score

B. Dataset

TABLE I. Dataset Description

Sr. No.	Dataset – u.data		
	Description	***Column***	***Type***
1.	ID of users.	user_id	int
2.	ID of items.	item_id	int
3.	Ratings of a movie.	rating	int
4.	Time of data generated.	timestamp	int

TABLE II. Dataset Description

Sr. No.	Dataset – u.item		
	Description	***Column***	***Type***
1.	ID of movie.	movie_id	int

Sr. No.	**Dataset – u.item**		
	Description	***Column***	***Type***
1.	ID of movie.	movie_id	int
2.	Title of the movie.	movie_title	str
3.	Movie release date.	release_date	datetime
4.	Release date of the video.	video_release_date	datetime
5.	URL of IMDB reviews.	IMDB_URL	str
6.	Movie Genre – Unknown	unknown	int
7.	Movie Genre – Action	Action	int
8.	Movie Genre – Adventure	Adventure	int
9.	Movie Genre – Animation	Animation	int
10.	Movie Genre – Children	Children	int
11.	Movie Genre – Comedy	Comedy	int
12.	Movie Genre – Crime	Crime	int
13.	Genre – Documentry	Documentary	int
14.	Movie Genre – Drama	Drama	int
15.	Movie Genre – Fantasy	Fantasy	int
16.	Movie Genre – Film-Noir	Film-Noir	int
17.	Movie Genre – Horror	Horror	int
18.	Movie Genre – Musical	Musical	int
19.	Movie Genre – Mystery	Mystery	int
20.	Movie Genre – Romance	Romance	int
21.	Movie Genre – Sci-Fi	Sci-Fi	int
22.	Movie Genre – Thriller	Thriller	int
23.	Movie Genre – War	War	int
24.	Movie Genre – Western	Western	int

TABLE III. DATASET DESCRIPTION

Sr. No.	**Dataset – Scrapped**		
	Description	***Column***	***Type***
1.	ID of movies from *u.data*.	movie_id	int
2.	Name of the movie.	movie_title	str
3.	Avg Sentiment of Movie.	sentiment_score	float

C. Matrix Factorization – Collaborative Filtering

Process of taking a matrix and decomposing it into a product of two triangular matrices [11].

Equation:

$$\sum_{(i,j)\in \text{obs}} w_{i,j}(A_{i,j} - \langle U_i, V_j \rangle)^2 + w_0 \sum_{i,j \notin \text{obs}} \langle U_i, V_j \rangle^2$$

where $w_{i,j}$ is a function of the frequency of query I and item j.

Example:

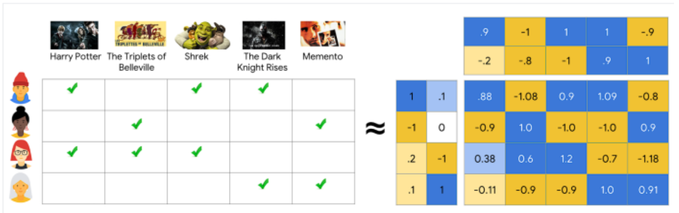


Fig. 2. Simple Embedding Model explaining concept

D. Neural Embedding Layer

A tool used to convert high-dimensional data into a lower-dimensional space.

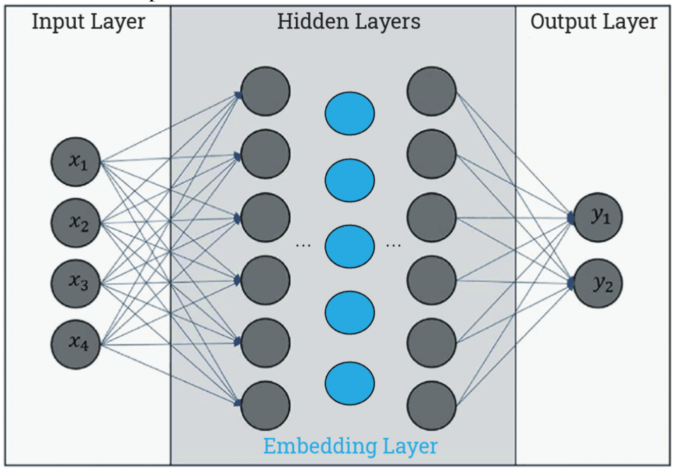


Fig. 3. Embedding Layer [13]

The layer converts data into vedctors. A vector is a list of numbers. Each words, assigned a unique vector.

E. Sentiment Score

VADER stands for Valence Aware Dictionary and sEntiment Reasoner), a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media.

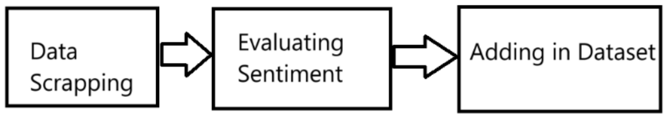


Fig. 4. Process Flow for Sentiment Score

While scraping the data from the web, the sentiment score is evaluated based on the comments fetched while scraping [14]. The VADER sentiment analyzer performs the operation, and an average of all sentiments is stored in the dataset for each movie.

F. Activation Function

Activation function calculates the weighted sum of inputs and adds a bias.

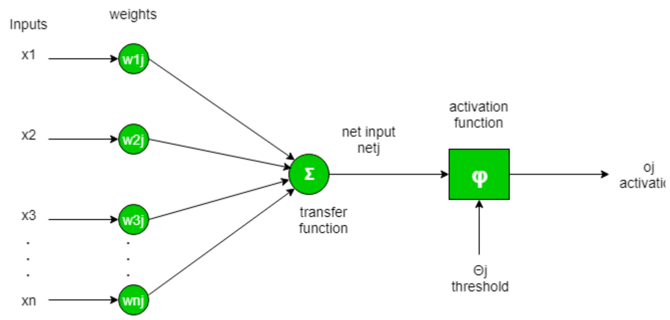


Fig. 5. Activation Function [15]

Formula: $\text{Net Input} = \sum(\text{Weight} \times \text{Input}) + \text{Bias}$

IV. Experiments

A. *Experiment Design*

In our project, we have tried X and Rotten Tomato for scraping data. While X was not allowing access to their data after a few trials, ending up with a few records incompatible for a neural network problem, the Rotton Tomato provided data with many null values randomly.

For the experiment, we have used the Latent Vector Size for Matrix Factorization and Embedding-Based Neural Network [2] for the hybrid model.

B. *Dataset Preparation*

The final dataset for the model training was split using the standard *train_test_split()* function with 80/20 ratio.

C. *Evaluation Metrics*

Metrics helps in identifying how well the model is performing. There are various metrics used in this process:

1) RMSE: Root Mean Square Error is average difference between model's predicted and actual values [2].

2) MAE: Mean Absolute Error helps in evaluating regressive model accuracy [5].

3) Precision: Number of true positive values either correctly or incorrectly identified by the model.

4) Recall: True positive rate measuring the true positives based on true positive and false negative values.

5) F1–Score: Derived from harmonic mean of precision and recall [16].

D. *Results and Analysis*

1) Hyperparameter Selection

Early Stopping: Stops the neural network to run more epochs based on the loss of the data detected while training to save the resources.

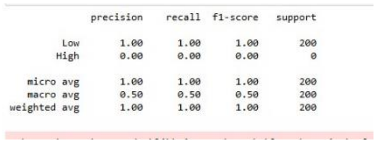
Learning Rate: Rate of learning to prevent vanishing gradient descent. We have used **0.001** as our learning rate.

Patience: Process of letting the epochs run if the early stopping parameter is not satisfactorily increasing/decreasing. We have used **5** as patience.

Optimizer: Optimization technique for gradient descent to prevent the vanishing gradient at a certain extent. We have used **Adam** optimizer.

2) Results of Recommendation Models

a) Collaborative Filtering



```
              precision    recall  f1-score   support

         Low       1.00      1.00      1.00       200
        High       0.00      0.00      0.00         0

   micro avg       1.00      1.00      1.00       200
   macro avg       0.50      0.50      0.50       200
weighted avg       1.00      1.00      1.00       200
```

Fig. 6. Metrics

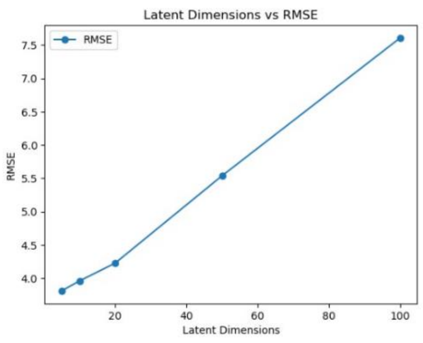
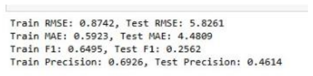
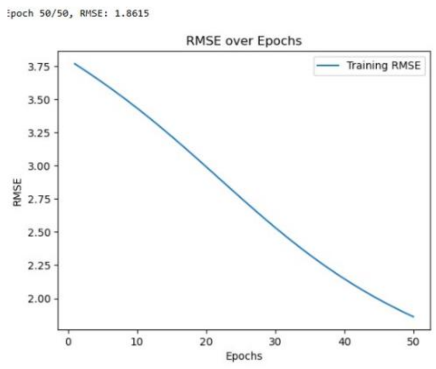


Fig. 7. Latent Dimensions vs RMSE



```
Train RMSE: 0.8742, Test RMSE: 5.8261
Train MAE: 0.5923, Test MAE: 4.4809
Train F1: 0.6495, Test F1: 0.2562
Train Precision: 0.6926, Test Precision: 0.4614
```

Fig. 8. Train-Test Metrics



```
Epoch 50/50, RMSE: 1.8615
```

Fig. 9. RMSE over Epochs Curve

b) Hybrid Model

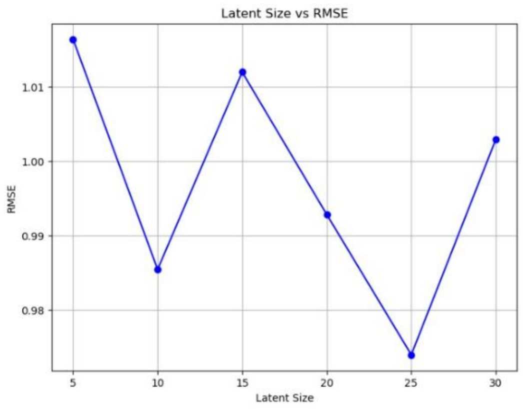
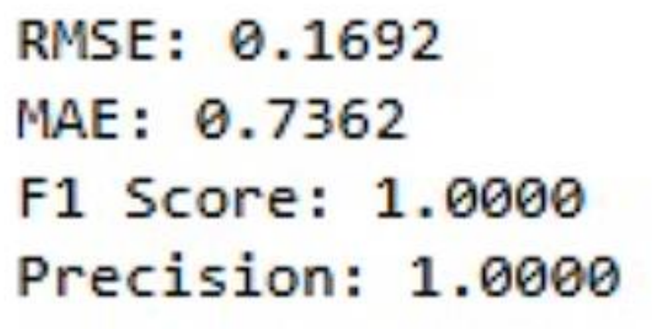


Fig. 10. Latent Size vs RMSE



```
RMSE: 0.1692
MAE: 0.7362
F1 Score: 1.0000
Precision: 1.0000
```

Fig. 11. Metrics

c) Model Comparison

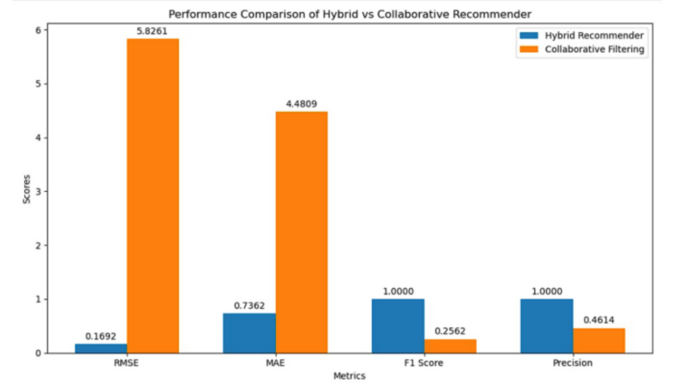


Fig. 12. Model Comparison

The Collaborative Filtering RMSE and MAE scores were significantly large as compared to the Hybrid Recommender. The Hybrid Recommender performs way better than the Collaborative model.

V. Conclusion

This project demonstrated the enhanced accuracy of hybrid recommender systems by integrating collaborative filtering with content-based features and sentiment analysis. The hybrid model outperformed traditional collaborative filtering across all evaluation metrics, validating the effectiveness of combining multiple data sources [1][2][5].

References

[1] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.S. Chua, "Neural collaborative filtering", in *Proceedings of the 26th international conference on the world wide web*, 2017, pp. 173-182.

[2] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems", *Computer*, vol. 42, no. 8, pp. 30-37, 2009.

[3] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering", *IEEE Internet Computing*, vol. 7, no. 1, pp. 76-80, 2003.

[4] A. Said and V. N. Mauro, "A comparative analysis of collaborative filtering algorithms", in *Proceedings of the 13th ACM conference on recommender systems*, 2019, pp. 215-223.

[5] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook", in *Recommender Systems Handbook*. Boston, MA: Springer, 2011, pp. 1-35.

[6] S. Funk, "Netflix update: Try this at home", *S. Funk's Blog*, 2006. https://sifter.org/~simon/journal/20061211.html.

[7] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms", in *Proceedings of the 10th international conference on World Wide Web*, 2001, pp. 285-295.

[8] M. Jamali and M. Ester, "Trust-aware recommender system", in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 983-992.

[9] R. Burke, "Hybrid recommender systems: Syrvey and experiments", *User Modeling and User-Apadated Interaction*, vol 12, no. 4, pp. 331-370, 2002.

[10] C. C. Aggarwal, "Content-based recommender systems", in *Recommender Systems: The Textbook*. Cham: Springer, 2016, pp.139-166.

[11] GeeksForGeeks. *What is Embedding*. Online. https://www.geeksforgeeks.org/what-is-embedding-layer/

[12] CalcWorkshop. *Matrix Factorization – Step-by-Step for* Students. Online. https://calcworkshop.com/matrix-algebra/matrix-factorization/

[13] baeldung. *What Are Embedding Layers in Neural Networks?*. Online. https://www.baeldung.com/wp-content/uploads/sites/4/2023/01/embedding_layer.png

[14] VaderSentiment. *Official Documentation*. Online. https://vadersentiment.readthedocs.io/en/latest/

[15] GeeksForGeeks. *Activation Functions*. Online. https://www.geeksforgeeks.org/activation-functions/

[16] GeeksForGeeks. *F1 Score*. Online. https://www.geeksforgeeks.org/f1-score-in-machine-learning/