

# **Multimodal Recommendation Engine with Feature Encoders**

City University of Hong Kong

Fung Cheuk Lam, Ace

Yau Kwun Fung, Kevin

Siu Him Kan, Steve

## 1. Motivation

The objective of this paper is to explore an equitable recommendation model for product promotions. Recommendation models are widely used in the movie industry by entities such as Netflix and YouTube Premium. These large entities have the advantage of accessing extensive user profiles, a luxury that small and medium enterprises (SMEs) do not possess. In many scenarios, these SMEs will rely on purchasing third-party data (cookies) to engage in data enrichment in hope to uplift their marketing and recommendation based services. However, cookies will be deprecated in 2024 (Moffett 2024) due to privacy issues and concerns, which causes immediate concerns for SMEs. As a potential solution, we are proposing a framework in this paper that provides a direction to train an in-house recommendation engine using first party data that is both effective and fair. With that premise, we will be using Movielens dataset in this paper.

In modern day recommendation engines for movies, accurate and fair recommendations could be rare at times. As an example, service providers like Netflix may promote movies based on their overall popularity rather than adopting a personalised approach. The recommendation model for Netflix is based on collaborative filtering and content-based filtering. It will collect the user's habits and movie content for modelling. Also, there is a promotion strategy called Netflix Originals that pushes exclusive content to the customer. Customers may tend to go for Netflix Originals and pay attention to those movies instead. Another prominent problem for movie recommenders stems from the imminent amount of paid and ghost writers posting inaccurate movie reviews with the goal of boosting traffic for a particular movie. To this end, the proposed framework in this paper will be opinion agnostic as an attempt to sort out the recommendation problems mentioned above.

In addition, recommendation engines are mainly built on top of user profiles and digital footprints, which is becoming harder to obtain for SMEs with the rising privacy concerns from the public and the commitment of Cookie deprecations by Google in 2024 [3]. To emphasise, we hope to demonstrate how SMEs can build an in-house multimodal recommendation using first party data in this paper using open source data. This should provide a general direction for entities that wish to promote their product based on other open source data such as OpenRice and Price.com.hk.

The primary contributions of this paper can be outlined as follows:

- Data analysis and preprocessing that contains data augmentations and feature engineering
- Experiments with non neural network based recommenders - Matrix Factorizations
- Incorporation of nonlinearity and encoders into Matrix Factorizations
- Propose dedicated encoders built upon pretrained CNN and NLP models
- Training optimizations and model architectures
- Future analysis and other use cases using item encoders for RAG chatbots

## 2. Related Works

Recommendation engines is a well researched topic in the field of machine learning. In the paper Matrix Factorization Techniques for Recommender Systems (Koren et al 2009) [1], the idea of matrix factorization is established for recommender systems. The model factorises the user and item interaction matrix, such as the rating matrix, into the product of two lower-rank matrices, where the rank is a parameter to be selected. This idea helps capture the low-rank structure of the user-item interactions. In the algorithm, it utilises quadratic optimization to factorise the interaction matrix into two latent factors. This model achieved a RMSE of 0.8563 and it played a vital role in winning the Netflix competition and becoming the standard of the recommender system. As such, we will use this as our baseline model for comparison.

In the paper Neural Collaborative Filtering (He et al 2017) [2], it utilises neural networks to enhance flexibility of the model and capture the nonlinearity in the user and item interaction. The model is composed of two neural networks. The first is named generalised matrix factorization (similar idea to the aforementioned Matrix factorization by Koren et al). The second is a neural network and models the nonlinear interactions from two pathways. They also found that with a deeper network, the performance also improves. Finally, the two submodels are concatenated together and regressed onto the output for scoring. This model is also very popular recently as it is able to capture more complex relationships in different datasets. The proposed framework achieved a hit rate of 72% using the Movielens dataset.

In the paper A Dual Augmented Two-tower Model for Online Large-scale Recommendation (Yu et al 2021) [3]. The two-tower model is designed to capture and learn embeddings of user and item separately. However one shortcoming of the model is that usually the two towers only interact at the final stage. The researchers created a Dual Augmented layer. In addition to the user rating tower, they concatenated additional features such as city and gender, and an Augmented vector to interact with the item tower. Augmented vectors are expected to contain information about the current query and historical positive interactions. Finally, the two towers pass through neural networks and interact twice. If the label is true on the item tower, it will write an Augmented vector to the user tower, vice versa. Finally, a dot product is computed on the two output layers and used for scoring. This will be referenced in our proposed model. We aim to use multiple data types such as user ratings, movie casts, posters, comments, etc. The concatenation of data into the two towers and allowing interactions are a good idea that we will learn from. This proposed framework outperformed all baseline models improving HitRate@100 by 4.84% and 6.63% on Meituan and Amazon dataset over the best baseline (Yu et al 2021) [3] in their experiments.

## 5. Data Selection and Feature Engineering

To build our recommendation module, we used the movie dataset provided by MovieLens. Their data schema is somewhat constrained primarily due to user data being Personally Identifiable Information (PII) that is not publicly accessible. Likewise, the original dataset

lacks comprehensive movie metadata, consisting only of movie titles and genres. For the purpose of this paper, which aims to demonstrate how to train a medium to light weight recommendation system, we sought to enhance the data modality. To this end, we have developed a custom web scraper to obtain movie descriptions, posters and information on cast members. Due to limitations imposed by HTTP timeout restrictions, our scraper was only able to obtain 50% of the available moviedata. Nonetheless, this subset has substantial size that can be mapped to over 1,000,000 user records.

Similar to the user side, we have created a watch frequency dense matrix representing the watching frequency of each user across 20 different movie genres. During training, this 1 x 20 matrix will be encoded with linear layers and activation functions to fine grain watching behaviours.




	Action	Crime	Romance	War
John 	5	1	3	5
Tom 	3	4	5	2
Alice 	4	5	3	5

Fig 5.1 Watch History Dense Matrix

To conclude, we present our final data schema as follows.

User Side	Item Side
<ul style="list-style-type: none"> <li>• Movie Rating</li> <li>• Movie Watched</li> <li>• Timestamp (review)</li> <li>• <b>History Dense Matrix</b></li> </ul>	<ul style="list-style-type: none"> <li>• Movie Names</li> <li>• Movie Genres</li> <li>• Genome Ratings</li> <li>• <b>Movie Descriptions</b></li> <li>• <b>Movie Posters</b></li> <li>• <b>Cast Members</b></li> </ul>

Fig 5.2 Data Schema for training sets

## 6. EDA

It is widely acknowledged that data sparsity presents a huge challenge to model training and performance. Common approaches such as class balancing and sampling are used to address this in particular, but we reckon that recommendation engines should still somewhat reflect the popularity element of certain items. Therefore, we will simply apply a rating filter such that the training subset with users who have made 1000 or more comments. This filter serves to address the cold start problems encountered by most recommendation engines.



Fig 6.1 Wordcloud for Movie Genres

For the ratings distribution, it is quite skewed to the highest scores. Most of the users give above average (2.5) rating to the movies. As in reality, people will not waste time on reviewing bad movies but promoting their favourite movies. Also, classical movies have the most ratings. Movies that are published earlier tend to have more reviews. And some movies like Star Wars have multiple episodes. We may consider combining them to give higher importance, but no augmentations were done to user ratings in our experiments.

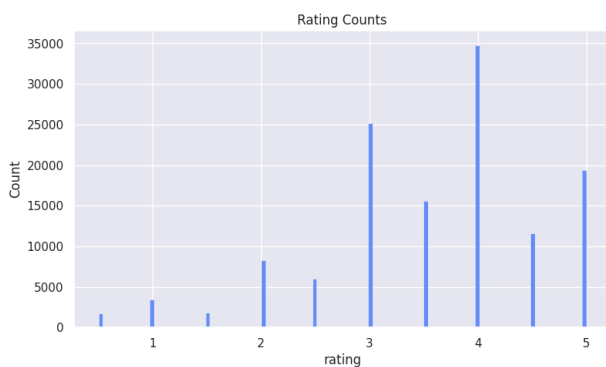


Fig 6.2 Ratings bar chart

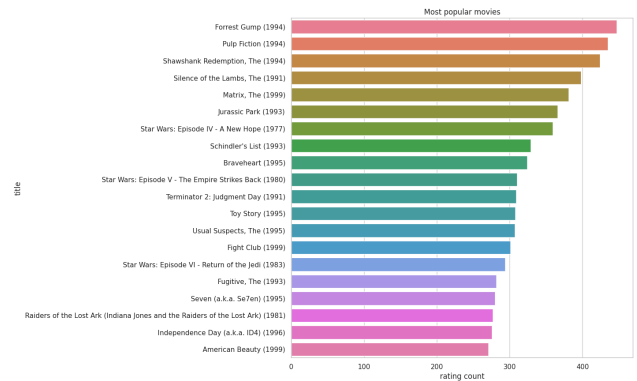


Fig 6.2 Rating bar chart by movies

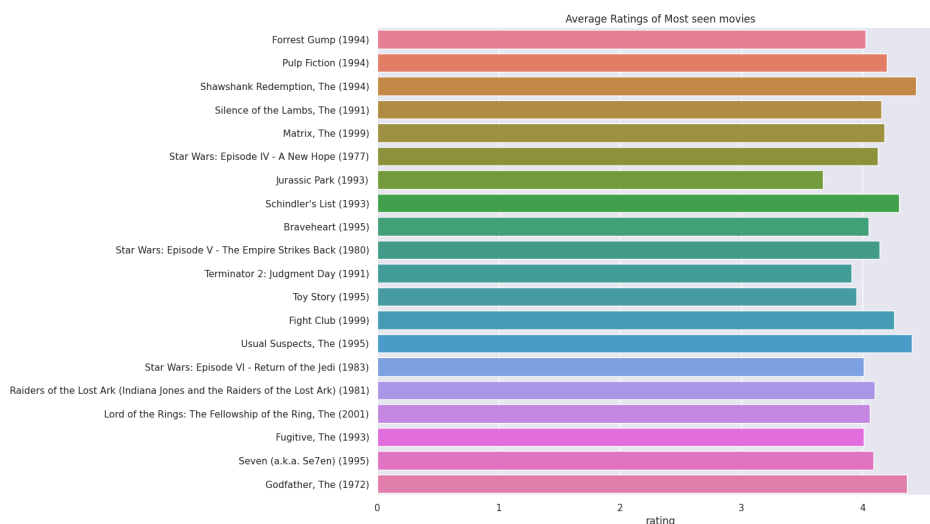


Fig 6.3 average ratings of top movies with highest reviews

In terms of non-tabular data, we also explored the inclusion of movie posters in the training process of our multimodal recommendation model. Movies within the same genre tend to employ similar colour palettes in their posters with the goal of conveying a specific message to their audience. For instance, movies with eerie and haunting themes typically use dark

tones, while lighter tones are commonly associated with romantic or comedic movies, as well as movies targeted at the younger viewers.

To capture these artistic preferences for each movie, we propose using Convolutional Neural Network (CNN) to train a dense vector representation. This representation takes into account not only the movie descriptions and cast but also its artistic decision for promotional materials. Moderate image transformations including auto augment policies, colour jitter and random rotations were used when constructing our training sets.



Fig 6.4 Movie posters from vastly different genres



Fig 6.5 Image Transformations

## 7. Other Design Choices and Data Augmentations

Pretrained embeddings from DistilBERT cased is used to encode the movie descriptions. Given that movie descriptions are generally short, we enforced the maximum length to be 64 during the tokenization process. Additionally, standard natural language processing techniques such as stopwords removal, punctuation removal and lemmatization were also applied to reduce data complexity and allow better model generalisations. The embedding

dimension of each description vector is a  $1 \times 768$  dense matrix. To reduce the overall trainable parameters, we implemented custom Multi Head Attention Layers (Encoder) instead of using the original pretrained BERT model in one of our models (MF\_NN).

Dimensionality reduction techniques like Principle Component Analysis (PCA) was initially applied on Genome Ratings, which is a  $1 \times 1128$  dense matrix. During our experiments, we explored various dimension settings, including reductions to 32, 64, 128, 256 and 512. However, we did not observe significant improvements in model performance. Thus, this augmentation was excluded during the training of all our models.

Other filters including training exclusively on users that have made over 1000 comments are applied throughout the training of all our models. In our experiments, we also trained with genome ratings instead of movie genres since the latter is a  $1 \times 20$  sparse matrix while the former is a  $1 \times 1128$  dense matrix. These design choices are attempts to avoid data sparsity.

## 8. Proposed Framework and Training Optimizations

Here we introduce the **Two Stage Encoder Model (TSEM)**, a recommendation framework comprising two distinct encoders for item and user features. This framework follows the common two tower approach to effectively capture item and user characteristics.

The item encoder consists of a series of dedicated linear layers for Genome Ratings. Non tabular features such as movie description and posters will be handled with pre-trained CNN (MobileNet\_v3\_s) and DistilBert Cased. Results from all layers will be concatenated before they are projected to lower dimensions and interacting with user profiles with a dot product mechanism. The dimension and number of hidden layers in our experiment varies, but they generally range from 128 to 512 and 2 to 6 respectively with ReLU as the activation function.

MobileNet\_v3\_s and Distil Bert Cased model have 2.5M and 110M trainable parameters respectively. Obtaining acceptable model performance with this proposed framework might require a sizable amount of training data and memory intensive compute engines. In our experiments, our initial training set is a small fraction of the millions of user ratings available. We also allowed different / more training data once the overall model loss achieved 0.61. This resembles the idea of fine-tuning. The machine used in our experiments has a RTX 4090 with 16 GB DDR4 RAM and a 8 Core / 16 threads CPU. Depending on the batch sizes and training volume, the training time for each epoch can range from 10 to 30 mins.

The user encoder consists of an embedding layer that trains with user ids. The embedding dimension is 256 / 512 in our experiments. User embeddings will interact with the item vectors from the item encoder, which ensembles the idea of Matrix Factorization. User frequency matrix will be concatenated with the results of the dot product instead. We based this design on our belief that the history of watched movies should be separate from item vectors and closely tied to user ratings.

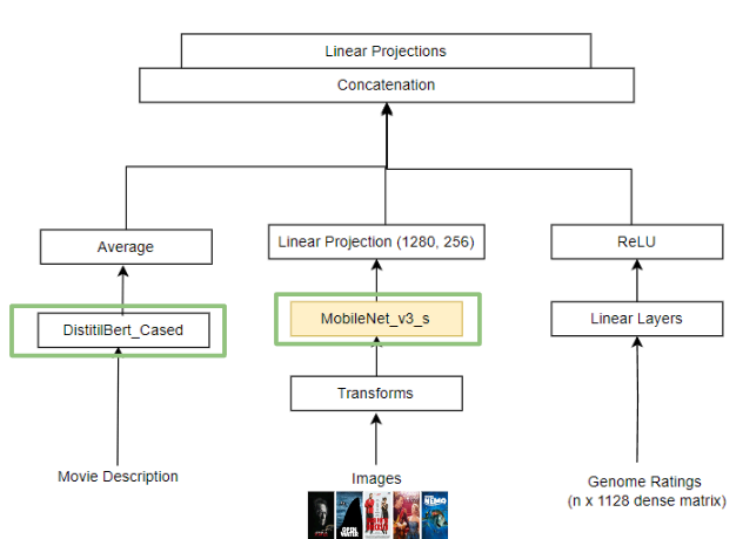


Fig 8.1 Item Encoder Architecture

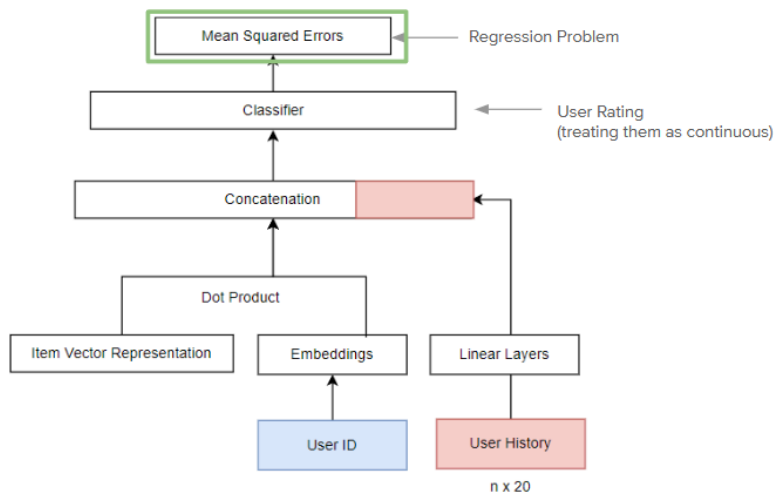


Fig 8.2 User Encoder Architecture

## 9. Models Comparisons

The dependent variable is user ratings and our models will be under the regression spectrum with Mean Squared Error (MSE) as the loss function. Our implementation of hit rate is an average accuracy across a random batch of users. The accuracy is calculated from a binary classification task with a predefined threshold ( $\geq 4$ ). Hit rate for each model **should not be** compared directly due to random sampling. Instead, this metric is merely used as a naive recommendation accuracy benchmark when given a threshold.

	a) NN	b) MF	c) MFNN	d) TSEM	(e) MF (Koren et al 2009)	(f) NCF (He et al 2017)
MSE	1.12	0.72	0.63	0.61	0.73	/
RMSE	1.06	0.85	0.80	0.78	0.86	/
Hit Rate	50.22%	73.08%	67.13%	73.88%	/	72%

### a) Simple Neural Network

A simple Neural Network with an user embedding layer and 2 linear layers with Relu activations and a classifier. This model trains on user ids and genome ratings of each movie. This model serves as the baseline model for benchmarking purposes.

### b) Matrix Factorization

A direct application with reference to Matrix Factorization Techniques for Recommender Systems (Koren et al 2009). It is the first paper that formally utilises matrix factorization on Recommendation system and is the foundation of all preceding models. Matrix factorization



helps capture the low-rank structure of the user-item interactions. In the training, we created an embedding layer for user and item, and used it to estimate the latent user-item matrix.

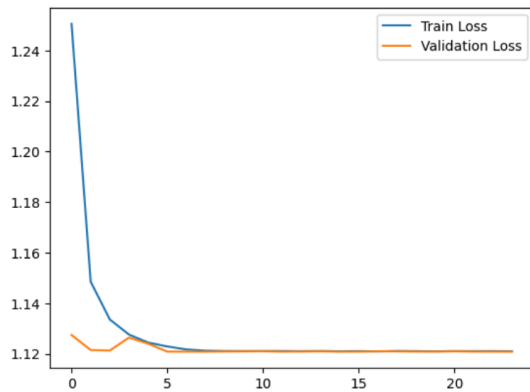


Fig 9.1 Train / Validation Loss for NN

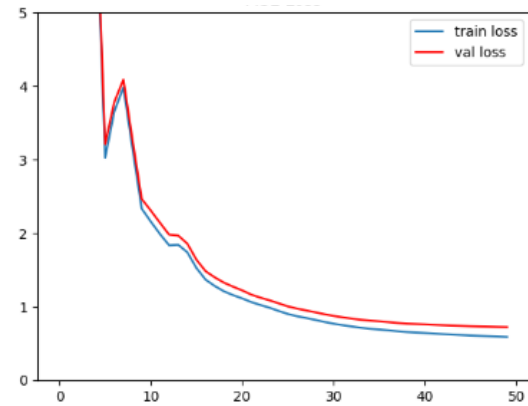


Fig 9.2 Train / Validation Loss for MF

### c) Matrix Factorizations Neural Network (MF\_NN) with textual data

This model is a modification from the Related Work 2 (Neural Collaborative Filtering). It utilises a two tower model framework, with the first being a classical matrix factorization. On the second block (item encoding), we included textual data such as Movie description and cast members. For the cast member, we used one-hot encoding to embed them as an input. For movie description, we used the pretrained Distill-bert-cased model's embedding. Then we concatenated the two movie-inputs and fit a classifier model with Sigmoid activation function. This serves as an analogy to movie genres. Lastly, the item block and user-item matrix are concatenated and passed through a neural network for the rating output

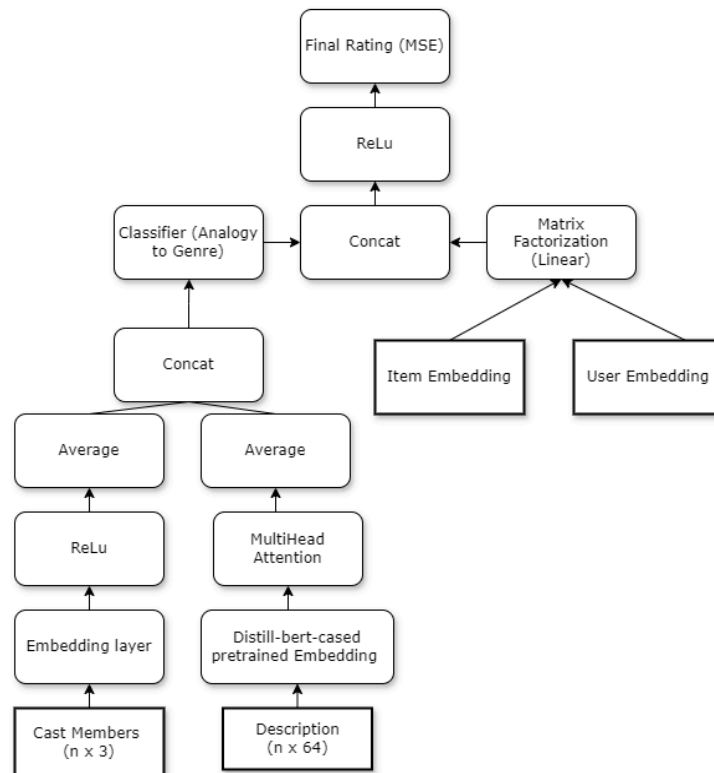


Fig 9.1 Model Architecture for MF\_NN

### c) Two Stage Encoder Model (TSEM) -Proposed Model

Our proposed model uses item and user encoders to train on movie images, descriptions, genres, user ids and user watch history. The interaction effect of item and user profiles is made from taking the dot product of the vectors. Please refer to section 8 for more information. CosineAnnealingLR is also used as our learning rate scheduler for the model with the maximum iteration steps set at 15, which explains why there is a sudden jump in validation loss in figure 9.4. As mentioned in section 8, we allowed more training data after reaching the same validation loss (MSE 0.61) as MF\_NN, which also happens at the 15th epoch.

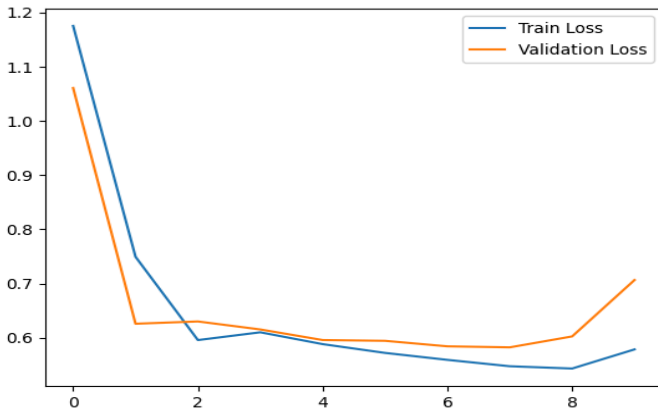


Fig 9.3 Train / Validation Loss for MF\_NN

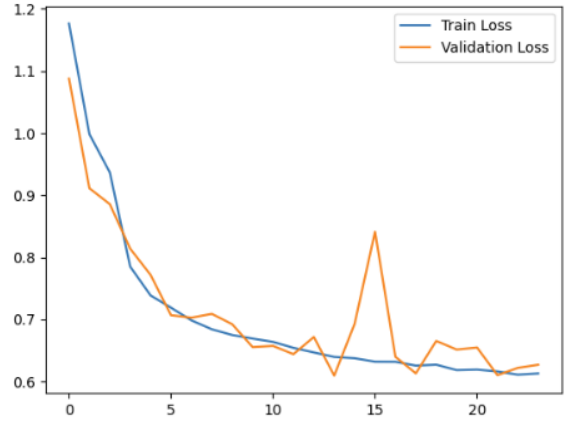


Fig 9.4 Train / Validation Loss for TSEM

The performance from our baseline model to the proposed model has progressively improved as we introduce more data modalities and model complexity. A simple Matrix Factorization can cut down the MSE by half from 1.12 to 0.72. Our incorporation of an item encoder training on cast members and movie descriptions into Matrix Factorization forming our MF\_NN model can further reduce the MSE to 0.63, which is at least a 10% increase in performance over our baseline models and the proposed framework by Koren et al in 2009 using Netflix data that is similar in data nature. Although we do not recommend a direct comparison of hit rates due to random sampling, MF\_NN has a 6% drop in positive recommendations if the classification threshold is set at 4. In terms of model loss, however, item embeddings with dedicated item encoders performed better than simple embedding layers in this experiment.

With that premise, we proposed TSEM with dedicated encoders that train with more movie related metadata to form better item representations. We also removed the item embedding layers in this framework. The MSE with this framework training with 30% of the data is around 0.61, which is similar to MF\_NN that was trained with at least twice the amount of user data. The hit rate@50 under TSEM with a subset of the original data reached 73.88%.

Due to resource restrictions, we cannot train with the same amount of data that was used with MF\_NN. Regardless, this demonstrated the strength of using encoders over simple embedding layers to construct item vector representations.

## 10. Conclusions

In conclusion, our choice of open source data and feature selections is intended to be opinion agnostic, with the goal of creating an unbiased recommendation engine. We also demonstrated that a recommender can be trained with limited user profiles.

This paper also presented a novel approach to build a recommendation engine by extracting latent features from both natural languages and images. We demonstrated that vanilla Matrix Factorization provides acceptable performance by simply using embedding layers to fine grain the interaction effect of item and users. The usage of dedicated encoders and the introduction of non-linearity and other modalities further improved model performance. Our proposed model with dedicated encoder layers training with movie images and descriptions demonstrated its advantage over using traditional embedding layers to construct item representations.

In future studies, we recommend training with more user data such as gender, age and other demographics related modalities if available. The interaction choice is not the core study of this paper, but we encourage future studies to make informed comparisons between dot product and other algorithms such as summation and concatenations.

Model optimizations such as LORA, adapters, word drop out embeddings and auto encoders can be considered to replace the fine-tuning procedures in our proposed framework. In the new era of LLMs, we also recommend using trained multimodal embeddings to create vectorstores for RAG chatbots. Retrieval performance using multimodal embeddings over natural language embeddings remains an interesting topic for future studies.

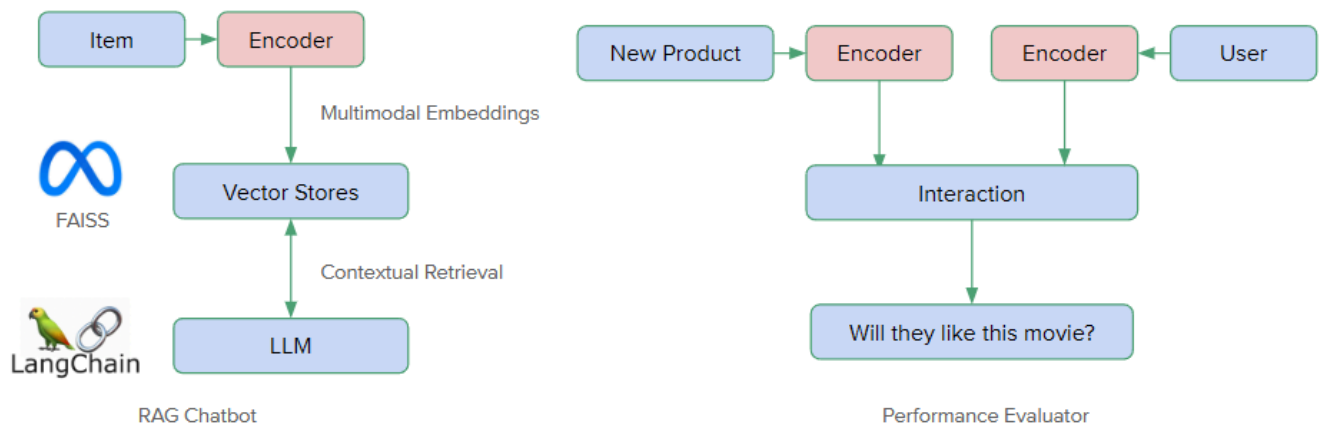


Fig 10.1 Other use cases with our proposed framework

## Citations

1. A dual augmented two-tower model for online large-scale recommendation. Y Yu, W Wang, Z Feng, D Xue. DLP-KDD, 2021. 11, 2021
2. Google Commits To Third-Party Cookies Deprecation In 2024. Tina Moffett. Jan. 2024
3. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T.-S. (2017). Neural collaborative filtering
4. Y. Koren, R. Bell and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," in Computer, vol. 42, no. 8, pp. 30-37, Aug. 2009, doi: 10.1109/MC.2009.263.