Movie Recommendation System : A Hybrid Approach

Group - 20

Under the co-guidance of

Dr. Vinay Joseph

(Assistant Professor, ECED)

Dr. Nujoom Sageer Karat

(Faculty Member, ECED)

Movie Recommendation Systems

- Recommender systems are an essential innovation by which content is curated by intelligently predicting user's interests and preferences.
- Recommender systems give us a plethora of relevant options which aid us in selecting the content to be consumed.

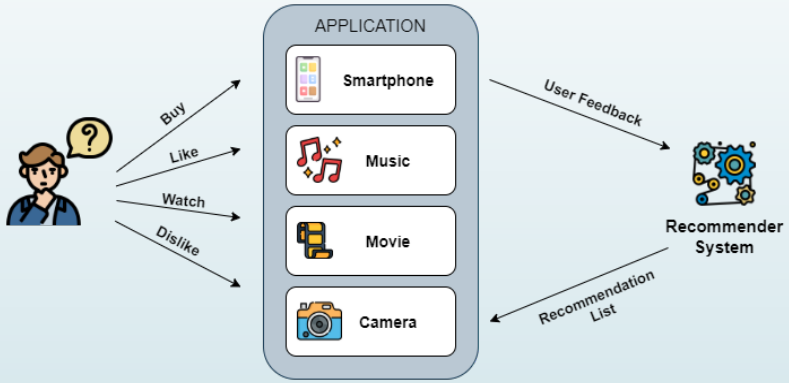


Figure-1: Application Scenario

- A Movie Recommendation System can suggest a set of movies to users based on their preferences or the popularities of the movies.
- This saves time for making decisions while consuming content for entertainment, especially in today's digital age.

Types of Recommender Systems

- There are broadly three classifications of recommendation systems in wide usage
  1. Content-based filtering
  2. Collaborative filtering
  3. Hybrid methods
- Hybrid approach is the integration of two or more recommendation algorithms like content-based filtering, collaborative filtering etc.

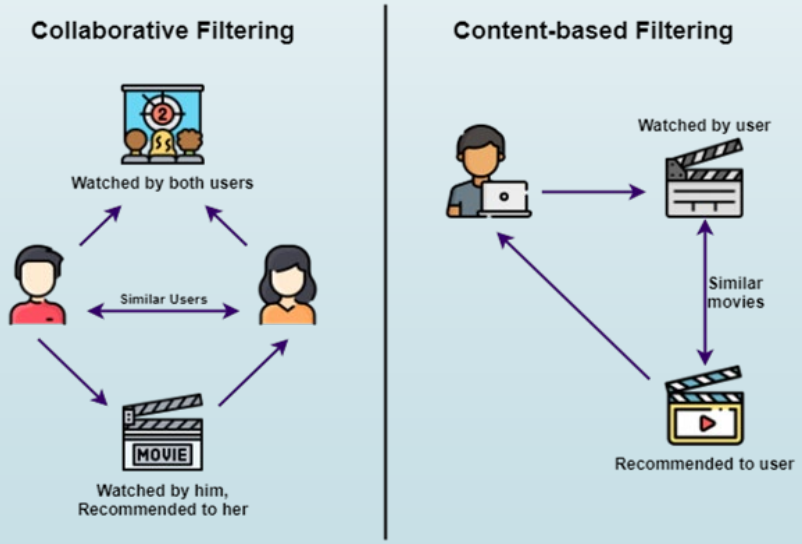


Figure-2: Types of Recommender Systems

Problem Statement

- To design and assess a novel movie recommendation system that utilizes a hybrid approach.
- To develop the hybrid approach using Matrix Factorization and BERT4Rec.
- To investigate the performance of our hybrid model using appropriate metrics.
- To tackle the disadvantages like lack of information about the domain dependencies in collaborative filtering and people's preferences in content based systems, with a hybrid approach.

Our Approach

- Pre-processing the dataset such that it meets our requirements.
- Design of Collaborative filter recommendation system utilizing Matrix Factorization.
- Design of Content-based filter recommendation system using BERT4Rec.
- Design of hybrid approach.
- Evaluating the performance of our model.

DATASET : ML-1m dataset was used, which included 1,000,209 anonymous ratings of approximately 3,900 movies made by 6,040 Movie-Lens users.

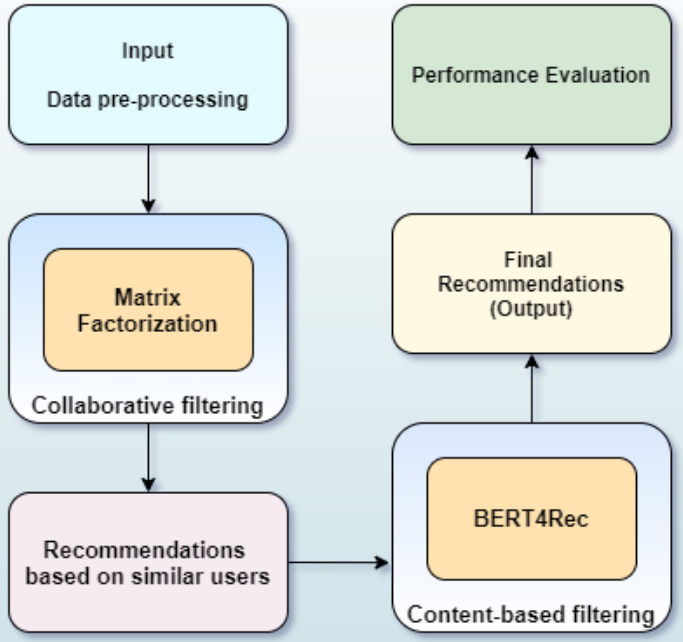


Figure-3: Abstract architecture of hybrid model

Collaborative Filtering

- Collaborative Filtering recommender systems create predictions based on user-item relationships.
- The user-item rating matrix is used in latent factor models for movie recommendations.
- A predicted rating for an item can be calculated by multiplying the factor vectors for the user and the object together.

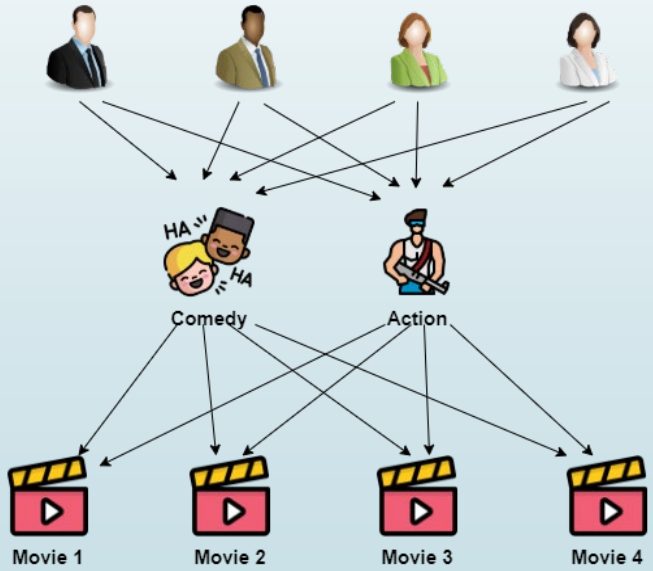


Figure-4: Collaborative Filtering

Matrix Factorization

- Matrix Factorization finds two rectangular matrices with smaller dimensions to represent a big user-item relationship matrix.
- The feedback (or rating) matrix is denoted as $A \in R^{m \times n}$, where m is the number of users (or queries) and n is the number of items.
- The model learns:
  - A user embedding matrix $U \in R^{m \times d}$, where row i is the embedding for user i
  - An item embedding matrix $V \in R^{n \times d}$, where row j is the embedding for item j
- We recommend by performing dot product of the factor matrices to fill in the missing entries in the rating matrix.

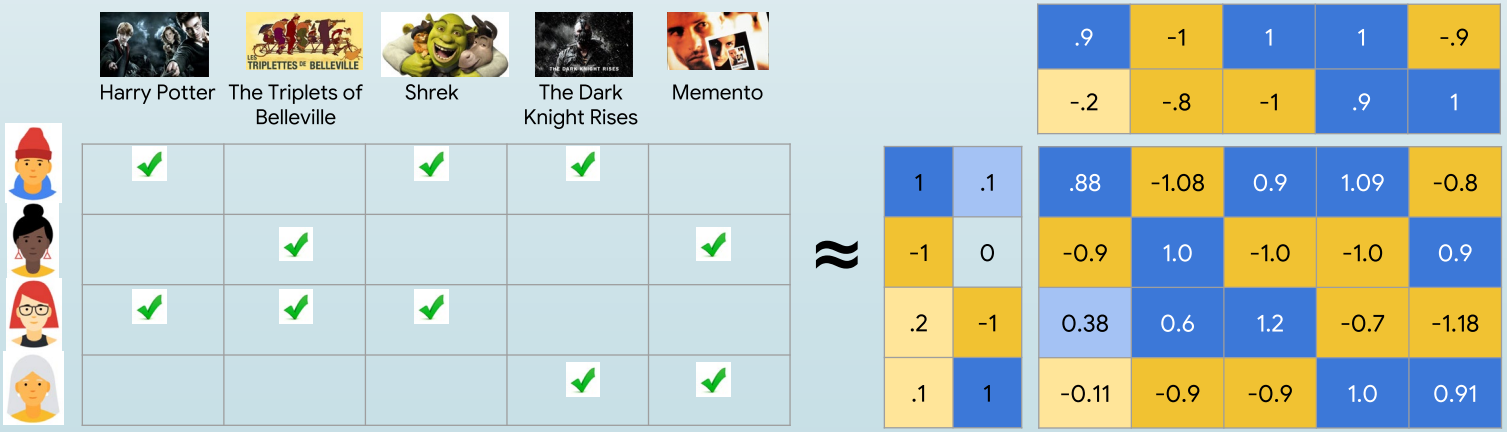


Figure-5: Matrix Factorization

Points from Last Meeting

Sparsity of the rating matrix

- The embeddings are learned such that the product UV^T is a good approximation of the feedback matrix A.
- Furthermore, the embeddings can be learned automatically, without relying on hand-engineering of features.
- Only sum over observed pairs (i, j), that is, over non-zero values in the feedback matrix.
- A matrix of all ones will have a minimal loss and produce a model that can't make effective recommendations and that generalizes poorly.

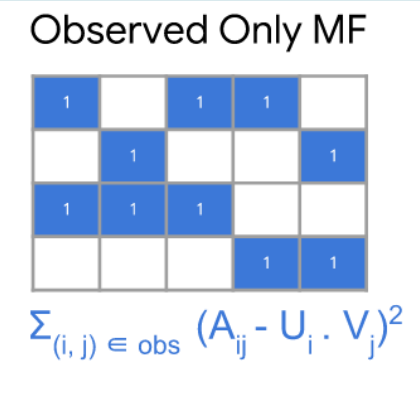


Figure-6: Observed Only MF

Points from Last Meeting

Evaluation of our model

- The train-test-split used is in the ratio 70:15:15 for the movie-lens 1m dataset.
- Along with calculating the training and validation losses, we have set aside 15% of our dataset to test our model.
- The metric used for testing is **Mean Squared Error (MSE)**
- MSE is a statistical metric that represents the standard deviation between a set of estimated values to the actual values.
- In recommender systems, it has been used to measure how far a set of predictions are, from the true values.

$$L = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \widehat{y_i}\right)^2$$

where,

N = number of data points

y_i = Observed Values

$\widehat{y_i}$ = Predicted values

Points from Last Meeting

Cosine Similarity – To verify the recommendations for user profile

- The movie name suggestions given by our model for a particular user are verified using Cosine Similarity, by comparing the predictions with highest rated movie of that user.
- Cosine Similarity is a measurement that quantifies the similarity between two or more vectors.

$$\text{cosine similarity} = S_C(A,B) := \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Where,

A – Weight vector of recent highest rated movie by a user

B – Weight vector of the movie recommended to a user

Results

- The MF model without bias has given a validation loss value 0.827 after training 100 epochs.
- MF model without bias has not given satisfactory results with validation data, which shows a diverging trend for generalization.
- The MF model with bias has given a validation loss of 0.754 after training 100 epochs.
- This model with bias has shown a good performance in training data and good generalization to validation data.

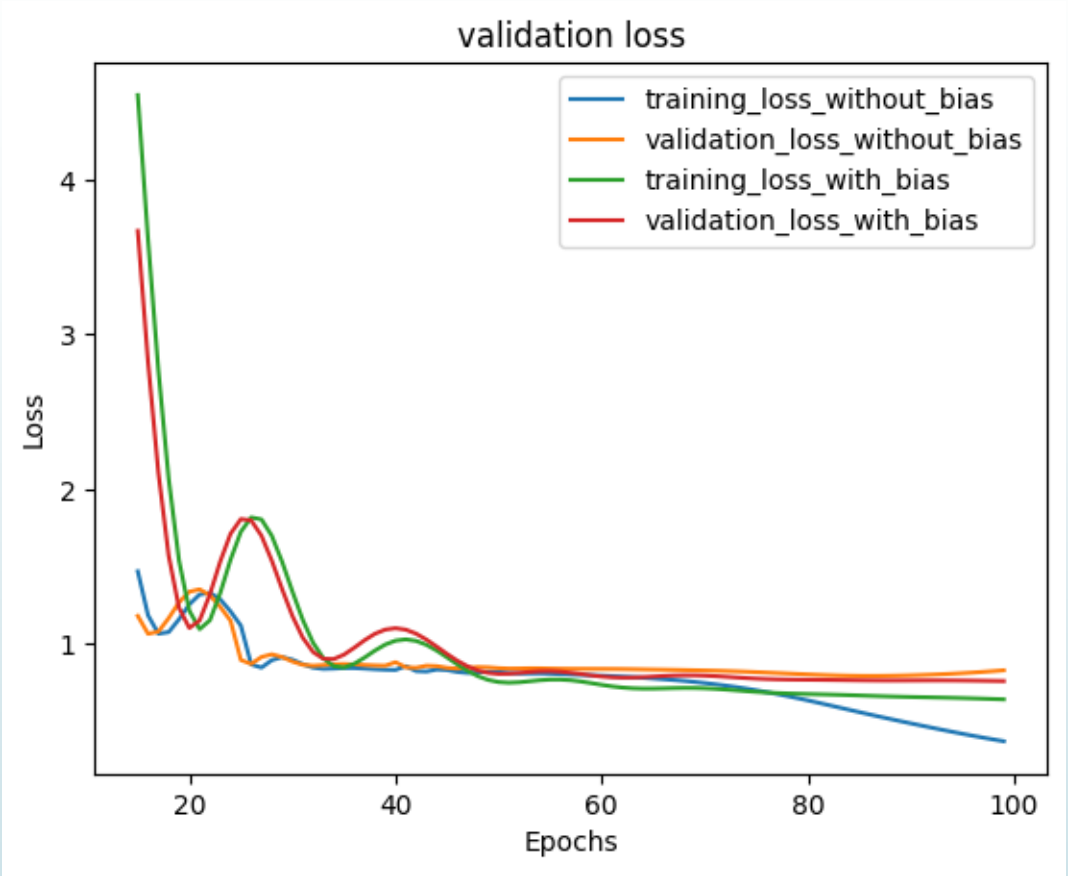


Figure-7: Training and Validation Losses vs Epochs

Results

- The Mean Square Error after testing the model is close to 0.75

```
[50] print(mean_squared_error(model,test))

     0.7568132776446861
```

Figure-8: Mean Square Error of Test data

- Using cosine similarity we found that the recommendations given by the model are similar to the recently highest rated movie by the user.

	userId	movieId	rating	timestamp	title
436658	123	435	5	975734584	Superman (1978)
693982	123	222	5	974668425	Kingpin (1996)
257307	123	377	5	974668380	Ghostbusters (1984)
231220	123	1618	5	974668351	Bowfinger (1999)
139234	123	533	5	974668323	Matrix, The (1999)
...	...	...	...	...	...
659068	123	259	1	974663031	Avengers, The (1998)
677463	123	1556	1	974663001	Super Mario Bros. (1993)
968582	123	2830	1	974662913	Fled (1996)
33651	123	698	1	974662023	Antz (1998)
880084	123	1604	1	974661959	Yellow Submarine (1968)

[281 rows x 5 columns]
Most recent high rated movie by the user : Superman (1978) 5

Figure-9: Movies rated by the user 123

```
Matrix, The (1999) --> 0.7347124
Star Wars: Episode IV - A New Hope (1977) --> 0.8711968
Raiders of the Lost Ark (1981) --> 0.9159394
Gladiator (2000) --> 0.7094488
Shawshank Redemption, The (1994) --> 0.7806139
Usual Suspects, The (1995) --> 0.7537628
Sixth Sense, The (1999) --> 0.8024993
Saving Private Ryan (1998) --> 0.7614989
Die Hard (1988) --> 0.88409793
Braveheart (1995) --> 0.7731308
```

Figure-10: Cosine distance between recommendations and Most recent highest rated movie by the user

Content Based Filtering

- Content-Based Filtering is a Machine learning technique that uses similarities in features to make decisions.
- This technique is often used in recommender systems, which are algorithms designed to advertise or recommend things to users based on knowledge accumulated about the user.
- BERT4Rec is the proposed algorithm to achieve content-based filtering.

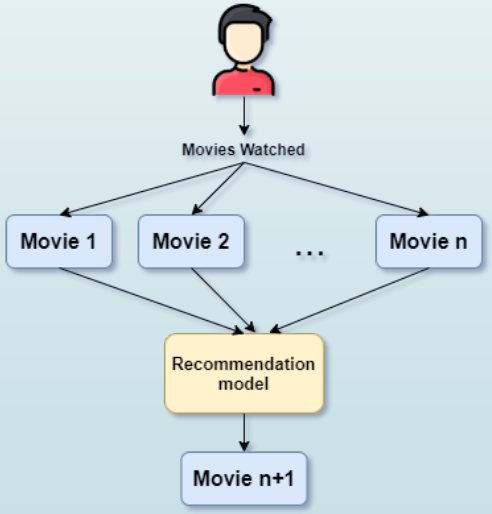


Figure-11: Content based model

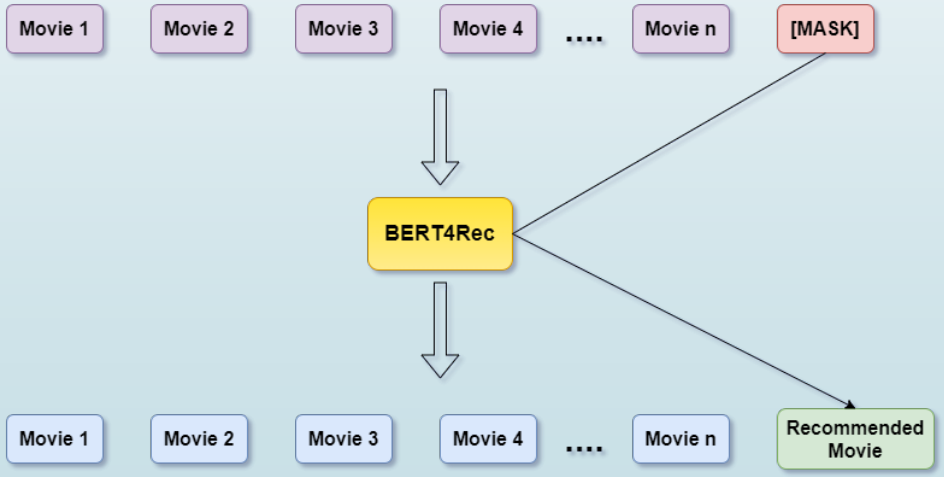
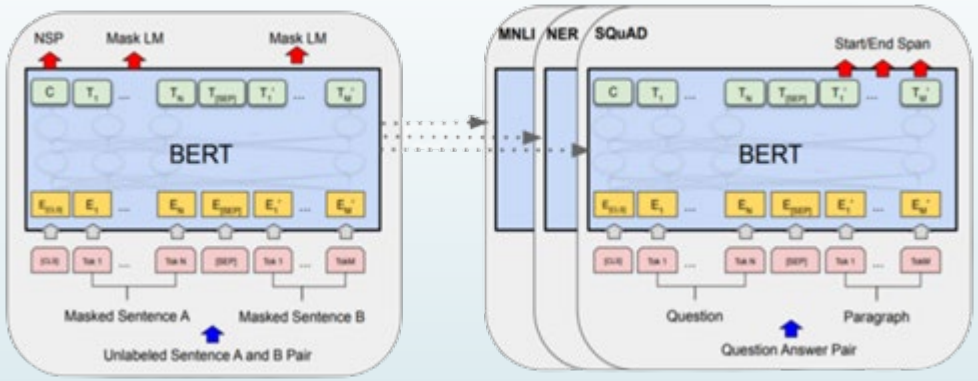


Figure-12: BERT4Rec model

BERT Overview

- Bidirectional Encoder Representations from Transformers (BERT) is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context.



Pre-Training Fine-Tuning

Figure-13: BERT Architecture

- BERT trained using two tasks :
  - Mask Language Model (MLM) : predict the masked words.
  - Next Sentence Prediction (NSP) : predict *IsNext or NotNext*

BERT vs BERT4Rec

BERT	BERT4Rec
• Commonly used in NLP tasks	• Used in Recommendation tasks
• Pretrained model that can be used for many NLP use cases.	• Model specific for a set of products in a platform
• Multitask learning (MLM and NSP)	• One task learning (MLM)
• Token represents word or sub-word.	• Token represents a product

BERT4Rec

- BERT4Rec uses bidirectional self-attention to simulate user's behavior sequences.
- We can easily capture item-item interactions across the entire user behavior sequence using self-attention mechanism.
- Transformer layer (Trm) is not aware of the order of the input sequence, without any recurrence or convolution module.
- In order to make use of the sequential information of the input, Positional Embeddings are added to input embeddings.

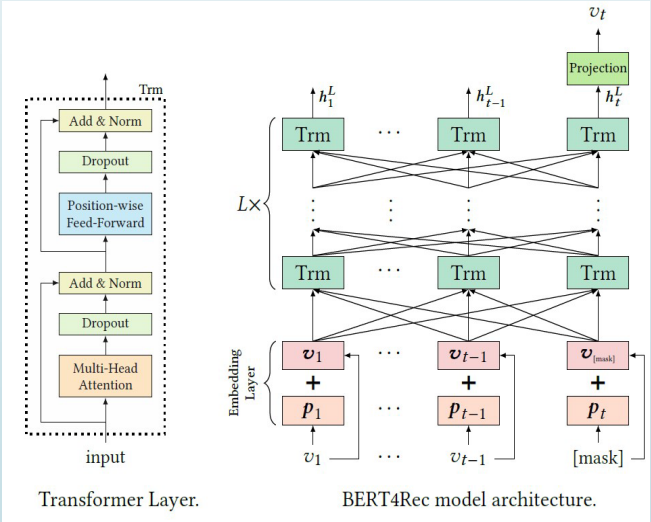


Figure-14: BERT4Rec model architecture

BERT4Rec Architecture

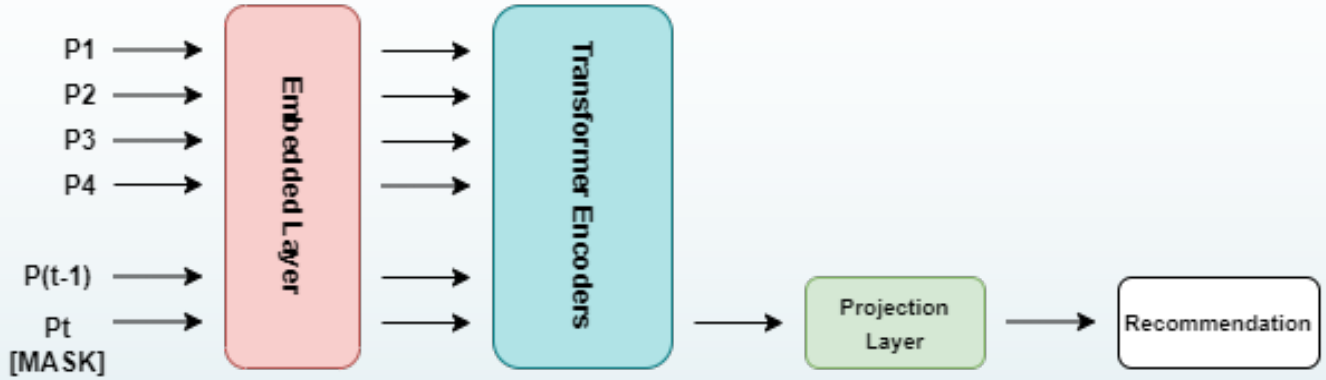


Figure-15: BERT4Rec block architecture

- Result of the embedding layer is the embedding vectors of the related products injected with learnable sinusoid positional embeddings.
- The encoder layer in the BERT4Rec model stacks multiple transformers layers to gather information from different position in the sequence.
- The projection layer is the final part of the model and it helps decide which product will be recommended.

Model Learning

Training

- For each training step, we randomly mask a proportion of all items in the input sequence, and then predict the original ids of the masked items based solely on its bidirectional context.
- Eventually, we define the loss for each masked input as the negative log-likelihood of the masked targets.

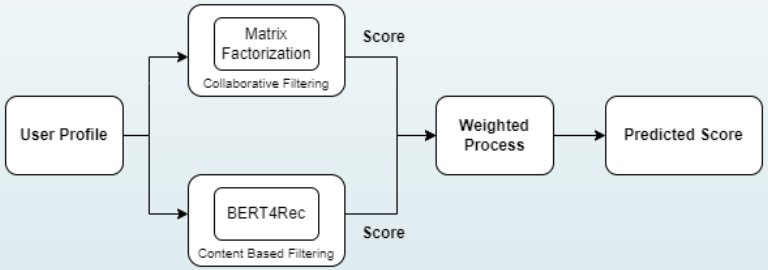
$$\mathcal{L} = \frac{1}{|\mathcal{S}_u^m|} \sum_{v_m \in \mathcal{S}_u^m} -\log P(v_m = v_m^* | \mathcal{S}_u')$$

Testing

- Append the mask token to the end of user's behavior sequence, and then predict the next item based on the final hidden representation of this token.

Future Work

- We plan to employ appropriate metrics such as Mean Average Precision to evaluate the BERT4Rec model.
- Weighted technique computes the prediction score as results of all recommendation approaches by considering them as variables in a linear combination.



Challenges

- BERT4Rec model needs high level resources for the model execution as the design is computationally complex.
- We have been using various free GPU sources like Colab and Kaggle, and we plan to purchase longer GPU sessions.

References

1) Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” Computer, vol. 42, no. 8, pp. 30–37, 2009

2) Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In The 28th ACM International Conference on Information and Knowledge Management (CIKM ’19), November 3–7, 2019, Beijing, China. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3357384.3357895

Thank you