Building a hybrid movie reccomender system with a state-of-the-art Transformer architecture

I. Introduction

This paper aims to critically evaluate the performance between personalised recommender system (PS) utilising state-of-the-art deep learning models and a popularity based non-personalized recommender (NP).

This paper adapts the works of Qiwei Chen et al. [1], who utilised a behaviour sequence transformer (BST) to provide E-commerce recommendations to user and applies it to help users find movies they will enjoy. The reason for choosing the BST [1] is its malleability and hybrid nature which combines Collaborative (CF) [3], content based (CBF) [5] approaches by concatenating several features which are fed into a pipeline consisting of a transformer [2] and multi-layer perceptron (MLP) neural network. Additionally, BST's ability to output predictions from short user histories make it ideal to tackle the cold start problem [6], one of the biggest real-world problems facing recommender systems. From this paper we explore the complexity of our PS and to compare it to our NP using evaluation metrics, which investigate the accuracy, diversity and serendipity of recommendations made. We also investigate the ethical concerns raised from our PS [1].

II. Dataset

A. Dataset description

The MovieLens1M dataset is used in this study. The dataset consists of 1,000,209 anonymised user ratings made by 6,040 unique users. The ratings are 0.5-5 range with 0.5 step size including the timestamp of when those ratings are made, on 3,883 unique movies, with 177 movies not being rated before. The dataset user features such as occupation category of user, age group and sex of user. These are useful for our model, as by considering users' demographic the model can make a more personalised recommendation. Although, there is an imbalance in the dataset with 4331 males present, compared to 1709 females, which could mean that our recommendations for females would not be as great for females, due to less data.

B. Data Preparation

After investigating review count distributions, and average number of movies rated per user (165), users and movies with less than 8 review counts were dropped to decrease dimensionality of dataset, and to prevent the cold start problem [6]. This ensures presence of enough movies in ratings sequence such that when the sequences are inputted to the multi-headed transformer, there is enough data for self-attention to capture insight about the input sequence. Although, we handle the cold start problem here, the BST handles the cold start problem in predictions. This is because during predictions, the model requires the length of sequences to be greater than one and less than 8, hence even if the user is new or doesn't have enough of a history, a good prediction can still be made using the BST. The ratings are sorted according to timestamps, and movie id and rating values are grouped according to the user id to get the data in the correct format to generate sequences of movies watched for each user.

C. Feature Selection and Data transformation

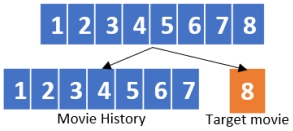

Figure 1 How input sequence is trained on

The model during training requires input sequences of fixed length, we split movie ids and ratings for each user into a set of sequences of fixed length 8 with step size as 1 to control the number of sequences we generate for each user. In a sequence of length 8, the first seven are training movie ids and the 8th is taken as the target movie as shown in Figure 1. The reason for choosing this was to generate the maximum number of sequences, whilst ensuring the sequences are not too short otherwise the model would not be able to capture sequential signals correctly. A total of 963,969 sequences are generated. Our model is considering user demographic, by embedding the user's sex, occupation and age group. The movie genres for each movie id are one hot encoded into vectors, to ensure they are in the right format, for embedding. From the genre and user demographic embeddings the model can establish an evolution of genres liked by a user by learning deeper relationships between the two.

III. Reccomendation techniques/algorithms

A. Personalised Reccomender System

Systems for commercial recommendations have frequently used deep learning-based techniques (RSs)[]. In previous works [7], raw features are projected to lower-dimensional vectors and inputted to a MLP for final recommendations. One such work being the Wide and Deep learning (WDL) [7] recommender system. WDL combines memorization and generalization, lacking in traditional CF, and learns deep relationships in data with memorization of simple rules. Although WDL is better than convention models such as CF and CBF still doesn't consider the sequential structure of a user's activity. This is where our model thrives – BST learns deep relationships between features and transformers which captures the underlying users' behaviors by using user features, movie features and user-movie-rating sequences. This results in more personalized recommendations for the user, even predicting the future movies a user would like.

B. Embedding layer

The first stage of the model requires the selected features to be embedded to a fixed-size lower dimensional space. In our

case, the size of lower dimensional space is chosen to be the square-root of the size of the vocab of the original feature consistent with the original authors [1]. In a transformer architecture, the embedding layers are necessary to create a continuous representation from the input sequence of tokens (sequence of movie ids in our case). To do this, a learnt embedding matrix is searched for the embedding vector for each token in the input sequence. The advantage of embeddings is that we can learn from large inputs and sparse vectors representing our features which is applicable to our domain. Furthermore, embedding layers captures semantics of the input sequence by placing similar inputs closer together in the embedding space. This is very useful because when embedding movie ids and user ids it gives the model an idea of which users and movies are alike, giving the model a CF aspect to it. Figure 2 represents the different features embedded, and some embedding spaces are multiplied, such as movie history with their corresponding ratings to scale the vectors of movies for each user by multiplying with its user specific rating.

Sequential and Positional Features	**Context Features**	**Content Features**	**Cross Features**
User id Target Movie id Movie history	User Sex Occupation Age group	Target movie genre Movie history genres	Movie history embedding * Movie history ratings Movie history embedding * positional encoding Target movie embedding * positional encoding

Figure 2 Embedded Features and which category they belong to

C. Multi-headed transformer

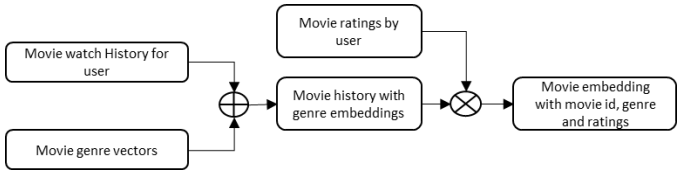

Figure 3 Non static features to embed sequences of movie ids for user

Figure 3 shows the pipeline for non-static features, which are inputted into the multi-headed transformer. Since transformers require sequences, static features such as sex and occupation not inputted. Movie history and target movies are examples of non-static features, figure 3 displays how we embed movies for each user. The pipeline first looks at the movie ids in the watch history and concatenates the movie ids with their respective one hot encoded genre vectors. To scale this vector, it is multiplied by the rating given to each movie id by the specific user. This generates movie embeddings which consider the user watch history, the genres they have watched and scale the vectors according to the user's ratings. This is inputted into the multi-headed attention block, which uses multiple self-attention mechanisms (eq.1) in parallel, where Q represents the queries, K with keys (movie ids) and V as the vector embedding of movie ids. This generates three matrices using linear projection which are then fed into the multi-headed attention layer in equation 3 to capture deeper relations with other movies in the users' behavior sequences.

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d}}\right)V \quad (1)$$

$$MultiheadAttention(Embedding) = Concat(head_i)\ \forall i\ \in [1,3] \quad (2)$$

$$Matrix = MultliHeadAttention(Embedding) \quad (3)$$

$$head_i = Attention(EW^Q, EW^K, EW^V)\ \forall i\ \in [1,3] \quad (4)$$

The Multiheaded attention outputs an overall representation for each users' behavior sequence, after concatenating the representation formed by each head in Equation 4.

D. Multi Layer perceptron

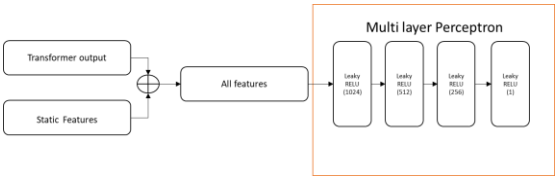

Figure 4 Architecture of MLP with output of Transformer layer

The transformer output is concatenated with the other features as shown in fig 4 and are inputted into an MLP. The MLP enhances the model with non-linearity and learns features in a more hierarchical perspective. This outputs a float for the predicted rating of the target movie id.

E. Non personalised reccomender system (NP)

For our NP, we use a popularity-based systems; we calculate the weighted rating metric, which provides priority to films that have been rated in higher amounts. This improves the confidence of recommendations by ensuring recommended movies meet a threshold. We determined a minimum threshold of 1033 ratings (92nd percentile) for this system.

$$Weighted\ rating = \left(\frac{v}{v+m}.R\right) + \left(\frac{m}{v+m}.C\right) \quad (5)$$

Equation 5 is used to calculate the weighted rating for each movie where v Is the number of votes, R is the average rating, m is the minimum number of votes required and C is the average number of votes across all movies.

IV. Implementation

A. Interface

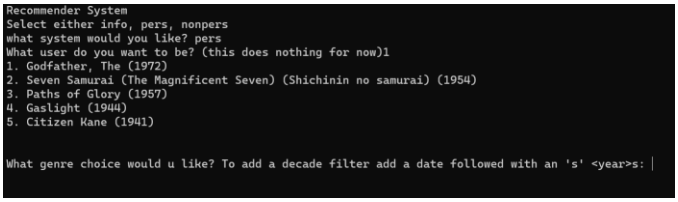

Figure 5 Interface and the recommendations made

Both NP and PS share the same interface. On system startup the user has 3 choices: 1) view privacy disclaimer 2) login to PS, and 3) access NP. The privacy disclaimer explains the data used specific to user by the PS. Following this, the user can login to PS by providing user id. With NP the user logs in anonymously. Once user accesses either system, the top 5 recommendations are displayed automatically. These recommendations can then be filtered based on genres and decade. E.g., for the 'War' genre movies from the 1990s, the user would have to input 'War' and '1990s' as an input. The filters can be cleared by inputting the string 'clear'.

V. Evaluation

To compare the NP/PS in this study, offline training and testing is used. To do this the data is split randomly into Train and Test splits of ratio 75:25 and stratified by the User ID, where in the test set, the ratings are kept hidden from the recommender. These experiments were repeated across 3 seeds, and the results were averaged. For assessing the accuracy of the rating predictions, predictions were generated for all movies in the test set and the mean-squared-error was determined between the predicted and actual rating. The reason for choosing MSE was because it penalizes larger errors, which in context of our domain has a greater effect than smaller errors, because larger errors impact the user experience more significantly than smaller errors.

To determine the relevance of recommendations of the systems, another metric used was the normalized-discounted-cumulative-gain as it considers both the relevancy of the recommended items and their rankings. nDCG compares the ranking of movies in the test set and the ranking according to the predicted ratings and considers that the most important items are recommended first. This is because top ranked movies are more likely to be interacted with the user, hence are key to the performance of the RS. Therefore, by ensuring that the top ranked movies are predicted correctly we ensure that the movies most likely to be interacted with the user are predicted correctly. Since nDCG is standardized, it is possible to compare it over various RS and datasets.

$$nDCG = \frac{\sum_{i=1}^{k} \frac{rel_i - 1}{log_2(i+1)}}{\sum_{i=1}^{k} \frac{rel_i^{truth} - 1}{log_2(i+1)}} \quad (7)$$

Finally, to evaluate the diversity of recommendations Novelty [8] is chosen as an evaluation metric. This is because it is crucial for an RS to promote new and different experiences to avoid boring the users with the same types of movies. Furthermore, if only the most popular movies are recommended, it could potentially result in the RS discouraging interaction with niche movies, and hence limiting user exploration and autonomy.

$$Novelty(R) = \frac{\sum_{i \in R} -log_2 p(i)}{|R|} \; s.t \; p(i) = \frac{|user\ ratings|}{|users|} \quad (8)$$

Metrics (on test set)	Personalised	Non-Personalised
MSE	0.73	1.45
nDCG@10	0.83	0.91
Novelty	2.72	1.31

Figure 6 NP and PS evaluation metrics

From fig. 6, the supremacy in accuracy displayed by PS is clear. Although PS does lack behind the nDCG score, which means that the ranking predicted by the PS is not accurate to the ground truth. The nDCG metric is favoured towards NP as it recommends the most popular movies. This means that NP performs better as popular movies are typically highly rated, hence NP would be better at predicting the ground truth ranking. The main advantage of the PS is that its recommendations are novel. This is reflected with the PS novelty score being almost twice that of NP. This is because NP regularly predicts the most popular movies, hence reducing diversity and preventing recommendation of niche movies. This is opposed to our PS, which models user behaviours and can predict the users' next interests resulting in more diverse and novel recommendations. This is very important as with greater diversity of recommendations we can expand the horizons of the user and prevent a convergence of possible recommendations.

VI. Conclusion

A. Limitations and future works

One of the limitations for our PS is the data preprocessing. The model requires sequences and a user rating history, which means that significant time is spent on processing the data into the right format for the model. Also, currently we are operating in the original space for all our features. One possible improvement would be extract latent features using an autoencoder and then train our transformer and MLP in the latent space, to improve training times. As mentioned previously, there is an ethical concern with our model, as we use user demographic information. This poses privacy risks, which risks the users' autonomy. Therefore, for the future, we could generate a mix of recommendations from the PS and NP. Also, currently there is no consideration of users' beliefs e.g., Religion, this could easily be embedded to prevent recommendations which may violate certain beliefs of users. Lastly, with the current configuration there is no way to add new users, which would need to be implemented to make this model operational in the real-world.

B. Overall remarks

In conclusion, our PS outperforms the NP in terms of accuracy and a greater novelty of recommendations. This is because our model considers the varying nature of users and models their interests over time to generate the best possible personalized results. Although there are concerns regarding privacy and decrease in user autonomy from our model. In the future the correct steps will be taken to mitigate these concerns as outlined above.

REFERENCES

[1] Behaviour sequence transfomer for E-commerce Reccomendation in Alibaba Qiwei Chen , Huan Zhao ,Wei Lei Pipei Huang , Wenwu Ou

[2] Attention is all you need Ashish Vaswani , Noam Shazeer , Niki Paramar , Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin

[3] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in ACM SIGIR Forum, vol. 51, no. 2. Page numbers:. 227–234

[4] ACM Transactions on Management Information SystemsVolume 11 Issue 2 June 2020 Article No.: 8pp 1–15

[5] H. -W. Chen, Y. -L. Wu, M. -K. Hor and C. -Y. Tang, "Fully content-based movie recommender system with feature extraction using neural network," 2017 International Conference on Machine Learning and Cybernetics (ICMLC), Ningbo, China, 2017, pp. 504-509, doi: 10.1109/ICMLC.2017.8108968.

[6] Blerina Lika, Kostas Kolomvatsos, Stathes Hadjiefthymiades,Facing the cold start problem in recommender systems, Expert Systems with Applications, Volume 41, Issue 4, Part 2,2014, Pages 2065-2073,

[7] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. , 7–10 pages.

[8] Solving the apparent diversity-accuracy dilemma of reccomender systems , proceedings of the National academy of sciences , 107(10) , 4511-4515, Zhou T. Kucsik Z. Liu J. G., Medo M. Wakeling J. R. & Zhang Y. C.(2010)