Leveraging Soft Clusters of Reviews for Personalized Product Rating

Manish Ranjan Karna
University of Massachusetts
Amherst, MA, USA
mkarna@umass.edu

ABSTRACT

Recently we have seen a lot of advancements in the field of Large Language Models (LLM). LLMs, for example, ChatGPT by OpenAI, are getting used for a variety of use cases including question-answer generation, paraphrasing, coding, etc. However, the content generated by these LLMs is not user-specific. Or to say, they don't provide personalized responses. This paper focuses on the personalization of the Large Language Model. We particularly use a data set by LaMP called LaMP-3U[5] consisting of product ratings given by users and try to personalize the ratings based on the user's experience. We propose a clustering-based approach to target the problem and compare our results with the benchmark set by the LaMP paper.

KEYWORDS

Large Language Models, Personalization, Prediction, Retrieval Models, Clustering, Soft Clusters

INTRODUCTION

Personalization of LLMs is needed to curtail the responses according to a user's behavior. Despite being a very versatile product these days, the LLMs still do not consider the user's experience and behavior to generate the outputs. Advancements have been made to make the LLMs more accurate in generating responses but these still lack the user's past to factor in the current response. Here, we study the effect of personalization in one of 7 datasets provided by the LaMP. The dataset is "Personalized Product Rating" user-based. It consists of the Amazon Review Dataset with some preprocessing already done like removing users with very less reviews as these are outliers for our use case. Our problem statement is to accurately predict the rating (between 1 to 5) given a review and user's profile. The profile consists of the user's rating of previous reviews. To target the problem, a simple solution is to feed all the existing reviews of the user to the LLM and get its output. However, this solution doesn't work because of the input constraints of the language models.

This leads us to reformulate our problem a bit. We need to selectively identify fewer number of reviews from the user's profile which could fit as the input of the language model. So, our primary focus in the rest of the paper is to effectively retrieve relevant profiles for a given review for a given user. This problem can also be seen as improving the retrieval model that was being used in the LaMP benchmark.

RELATED WORK

Personalization in Natural Language Processing (NLP) has been studied in the research community extensively. Existing review rating research mostly focuses on sentiment analysis and at times ignores the user's experience/past reviews and product information which are of considerable importance in predicting the rating. Personalization in sentiment analysis based on publicly available data has been studied in (Mireshghallah et al., 2022; Zhong et al., 2021). The approach followed here produces personalized responses by pre-pending a user-specific id to the input text. Different kinds of id has been experimented like alphabetic, alphanumeric and random user id. It turns out that a random user id performs the best. The user id is not part of the training. Most of the available personalization techniques uses a two-step process. In the first step, a generalized model is trained and in the second step, personalization is being done to cater the results for a particular user. However, the issue is that the model needs to be separately trained for each user in the second step. So, the paper proposes a single model that captures user's specific knowledge from the user id token. Another research focussed on personalized review generation (Li & Tuzhilin, EMNLP-IJCNLP 2019). Here, they propose a novel model to generate reviews for online websites like Amazon, and TripAdvisors by taking into account the product information and user behavior. The solution proposed uses auto-encoders and personalized decoders. The experiments carried out using multiple public datasets show that their model consistently outperforms other models and the results are in a similar style to the original reviews. However, our problem is to rate the product given its ratings and it does not involve text generation.

A very close research as our problem statement was done in (Wang B, Xiong S, et al., 2018). The research focuses on Review Rating prediction using User Context and Product Context. They proposed a model consisting of mainly 3 parts. The first part

learns a global review rating system, the second part learns a user specific review rating prediction and the third part is a product-specific review rating prediction model. This is a very good way of learning different aspects one by one, but in our case we don't have product information in the LaMP dataset and so, our experiments differs from this. Another research related to review rating has been carried out (Wararat Songpan). However, the focus here is based on the fact that some reviews and ratings don't align with each other. This means that other users can't infer much from the ratings provided, because there are data points where even though the rating is bad but the review is quite good. So they use different machine learning techniques like Naive Bayes and Decision Trees to predict the ratings. Also, the dataset used here is based on hotels reviews. But the proposed solution certainly lacks personalization which is the main factor in our research.

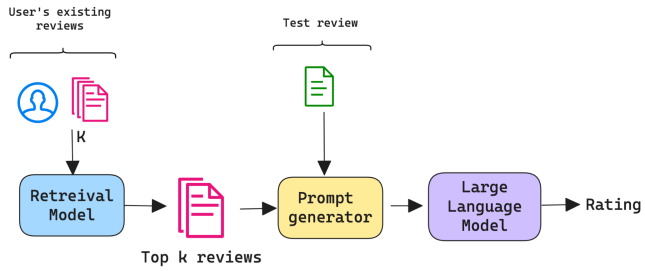


Figure 1: **Overview of personalized rating prediction**

PERSONALIZING LLM OUTPUTS

1 Problem Formulation

The dataset at hand consists of reviews and the ratings provided by the user in the past. In LaMP, we have two kinds of datasets for each of the problem types. The two kinds are based on user division and time division. In our research, we focus on the user division dataset. Here, the training, validation, and testing dataset have a non-overlapping set of users. Or to say, each user appears in utmost one of the three datasets. The goal is to finally predict the rating of a product based on the review written by the user, given the user's existing reviews/profile. However, as discussed above, we focus only on the retrieval aspect of this problem. Let the profile for a user u be represented as $P_u = \{(x_{u1}, y_{u1}), (x_{u2}, y_{u2}), \ldots (x_{un}, y_{un})\}$. This means that for the user u we have n existing reviews available. (x_{u1}, y_{u1}) denotes the pair of reviews and its respective rating from the user profile. Our problem statement is to effectively choose k pairs from the P_u which can be fed to the LLM as context to personalize the output. Thus we need to build a retrieval model M which takes input as P_u and a test review, and returns a relevant set of k pairs of reviews and ratings.

2 Clustering based approach

We propose a solution based on clustering of the user's existing reviews. First, we provide some insights about the dataset. The dataset contains the reviews and ratings per user. Each of the ratings is an integer between 1 and 5 (both inclusive). The maximum and minimum number of ratings for a given user is around 1000 and 100 respectively in each of the training, testing, and validation datasets. Since we have 5 different types of ratings, we create 5 clusters. Each cluster contains reviews of the same type of ratings. Formally, we have clusters namely C_1, C_2, .., C_5. Cluster C_i contains reviews having rating i $(i = 1, 2, 3, 4, 5)$.

The underlying idea behind clustering lies in the observation that users typically employ a limited vocabulary when crafting reviews. For instance, while there are over 20 synonyms for the term 'good,' a specific user's vocabulary might be confined to around 5. In our context, this implies that the words a user uses to articulate a positive review are likely to be frequently employed in other positive reviews. Therefore, by aggregating all past reviews of a user and grouping them according to their ratings, we may identify shared terms within each group. This gives us a sense to move towards clustering of the reviews on the basis of ratings.

The first step in our solution is to get the clusters created as explained for a given user. The point to note is that the clusters are created at a user level. For each user, we have a different set of 5 clusters. Next, for a given query review, we find the best clusters to which this query should go in. At this point, we employ soft clustering, which means we don't bind a review to a single cluster. Instead, we find the topmost clusters in which the review could go in. Formally, we find k clusters that mostly resemble with the test query. To do this, we first find the probability of each of the 5 clusters given the query review and then take the top k clusters. Thus, if the probability of review r going to cluster C_i is p_i, then we have the following condition:

$$\sum_{i=1}^{5} p_i = 1$$

$$\text{where } p_i = p(C = C_i \mid r)$$

To get the probability of a cluster given a review, we see the similarity of the review with the existing reviews in that cluster. Precisely, we first assign scores to each of the clusters. The score of a cluster shows how much relevance is the test review with this cluster. To get the score for a cluster, we first transform each of the existing reviews and test review to respective embedding

vectors and then we find the average cosine similarity of all the pairs of embeddings of existing reviews and the test review for a given cluster. The higher the score, the more chance that the test review should go into that cluster. The probability p_i is directly proportional to this score, the only difference is that the sum of all probabilities should be 1. When the score of each cluster is calculated, we find the top k clusters with the highest score. From each of these top k clusters, we get one most similar review. Thus at the end, we have top k reviews, which is considered as the output of the retrieval model M. These k reviews will now be used as the input to the LLM. These k contexts are expected to give the LLM an idea of how the user has been rating its reviews in the past. However, we don't focus on the results of LLM in this paper. Choosing the top k clusters diagramatically is shown in the figure 1.

EXPERIMENTS

1 Setup

Overall, our setup is that we first get the top k reviews for a given test review from the retrieval model M. Then these k reviews along with test review is fed to the Large Language Model. First, we focus on the retrieval model part. The retrieval model at a very high level assigns score to each cluster, takes the top k cluster and then takes one reviews from each of those k clusters. The score of each cluster is calculated independently. So, we can focus on how to calculate the score for one cluster. To assign the score to a cluster, we need the embeddings vectors corresponding to reviews. For this we have used the SentenceTransformer's (all-MiniLM-L6-v2) model to get the embeddings corresponding to a review. In the first step we find embeddings of test review and all the existing reviews of a cluster. Now we use cosine similarity between test review embedding and one of the existing review embedding to get a similarity score. We calculate such similarity scores for all the pairs and take its average. This average score is assigned to the cluster and we call it as score of that cluster. After the scores of each cluster is found, we choose the the top k cluster and then takes one reviews from each of those k clusters. Calculating scores for a cluster is shown in the Figure 2.

To get the final rating for a given test review and user profile, we have used the "Flan-T5 Small" model as the Large Language Model. Although in the LaMP paper, "Flan-T5 Base" is getting used, but using that is resource heavy, so we switched to the small model. Since, we're using "Flan-T5 Small" model, we're creating our own baseline score using BM25 as the retrieval model as it is being used in the LaMP paper. After getting the top k relevant reviews from the retrieval model M as shown in Figure 1 and described above, we transform the test review and top reviews as follows. Consider $PPEP$ as $'P_i[score]\ is\ the\ score\ for\ "P_i[text]"'$, then we have final string as: $concat([PPEP(P_1), ..., PPEP(P_n)], ",\ and\ ").[INPUT]$.

Here, $INPUT$ is the test review. Basically we are concatenating reviews with scores so as to get a final string which can act as input to the FlanT5 model. This is how we augment the prompt for input to the language model. The FlanT5 model has been used as it is in its pre-trained publicly available mode. We have chosen the validation dataset for the purpose of this paper, as we have validation ground truth outputs as well as this dataset is of considerable size and it doesn't require GPUs to process. The score given by the LLM is considered as the final rating, without modifying anything.

2 Results

The output of the LLM is an integer between 1, 5 (both inclusive). Given that our task is a multi-class classification problem, we use Root Mean Squared Error (RMSE) and Mean Squared Error (MSE) as the evaluation metrics. Since, the language models can't take as input very large list of tokens, so we have set the maximum input and output lengths as 512 tokens. We run our model for two different values of k, where k is the number of relevant reviews that the retrieval model is producing as output. The scores are given in Table 2. Our baseline performance is done using BM25 as the retrieval model with k = 1, as it's followed in the LaMP paper. The baseline results are RMSE of 1.04 and MAE of 0.72. We experimented with both $k = 1\ and\ 2$, but since our baseline is with $k = 1$, we compare only for k = 1. For RMSE, our model performs bit better than baseline with RMSE = 0.6848. There is not much improvement in MAE as our model's standing at MAE = 1.0464 and the baseline MAE is 1.0474. The comparison with baseline model is shown in Graph 2. We can also see that for $k = 2$, the results are bit worse than k = 1. This is shown in Graph 1.

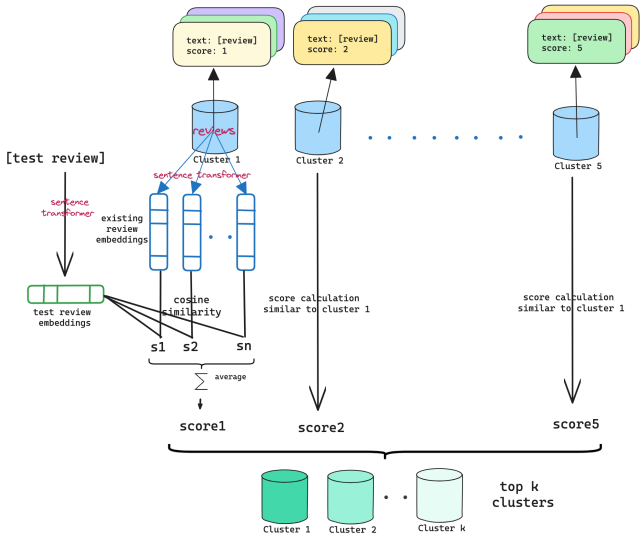


Figure 2: **Choosing top k clusters**

RESEARCH PROBLEMS

The analysed problem is of predicting the personalized rating for a review given the user's existing reviews. The solution discussed so far consists of finding the soft clusters in which the new rating could lie in and then outputting our results by taking into account the most similar reviews from the top k clusters. However, the solution doesn't consider the product for which the rating is being predicted, which is an essential factor to take in to improve our results. In that case, we can re-consider our way of how to do clustering maybe do on a per product basis. This product information was not considered as this information is not present in the LaMP-3U dataset.

Another direction to do research in this problem is to modify the template of the prompt passed to the LLM. The solution discussed just uses concatenation of the top k reviews. However, we can do some pre-processing about the user and come up with some average score that a particular user gives in general to any product based on past reviews and factor this generic score in the LLM output. This could be particularly useful in cases when we know beforehand that a user, no matter how good the review is, it is known to give utmost 4 rating, then next time as well for an excellent review, chances of 4 rating is quite high.

	Min #Reviews	*Max #Reviews*	*Length*	*#Classes*
Training	99	1028	20000	5
Testing	99	1006	2500	5
Validation	99	1023	2500	5

Table 1: **Lamp 3U Dataset Statistics**

Dataset	*Metric*	*Baseline* (BM25)	$k = 1$	$k = 2$
LaMP-3U: Personalized Product Rating	MAE	1.04785	1.04646	1.05311
	RMSE	0.72	0.6848	0.7004

Table 2: **Results**

REFERENCES

[1] Fatemehsadat Mireshghallah, Vaishnavi Shrivastava, Milad Shokouhi, Taylor Berg-Kirkpatrick, Robert Sim, and Dimitrios Dimitriadis. 2022. UserIdentifier: Implicit User Representations for Simple and Effective Personalized Sentiment Analysis. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3449–3456, Seattle, United States. Association for Computational Linguistics.

[2] [Towards Controllable and Personalized Review Generation](https://aclanthology.org/D19-1319) (Li & Tuzhilin, EMNLP-IJCNLP 2019)

[3] Wang, B.; Xiong, S.; Huang, Y.; Li, X. Review Rating Prediction Based on User Context and Product Context. Appl. Sci. 2018, 8, 1849. https://doi.org/10.3390/app8101849

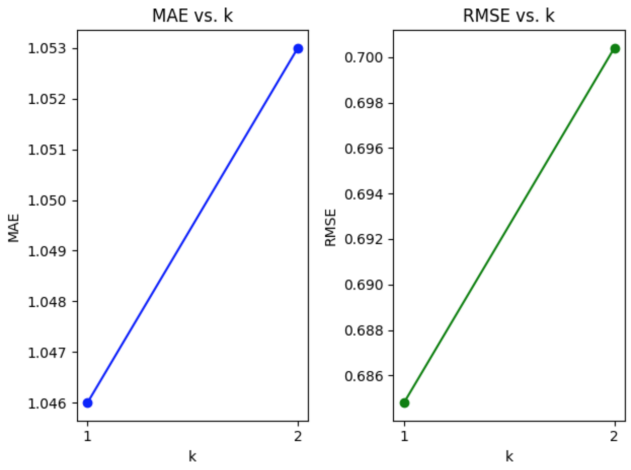
[4] W. Songpan, "The analysis and prediction of customer review rating using opinion mining," 2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications (SERA), London, UK, 2017, pp. 71-77, doi: 10.1109/SERA.2017.7965709.

[5] Alireza Salemi, Sheshera Mysore, Michael Bendersky, Hamed Zamani, "LaMP: When Large Language Models Meet Personalization", https://doi.org/10.48550/arXiv.2304.11406

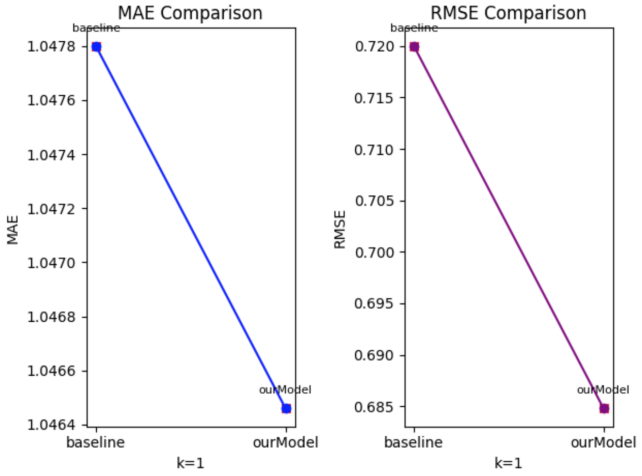
[6] Reimers, Nils and Gurevych, Iryna, "Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing", https://arxiv.org/abs/1908.10084

[7] https://huggingface.co/tasks/sentence-similarity

[8] https://pypi.org/project/rank-bm25/



Graph 1: **Results for our model (MAE, RMSE) for k=1,2**



Graph 2: **Comparison of baseline and ourModel (MAE, RMSE) for k=1**