MCM: A Multi-task Pre-trained Customer Model for Personalization

RUI LUO* and TIANXIN WANG*, Amazon LLC
JINGYUAN DENG, Amazon LLC
PENG WAN, Amazon LLC

Personalization plays a critical role in helping customers discover the products and contents they prefer for e-commerce stores.Personalized recommendations differ in contents, target customers, and UI. However, they require a common core capability - the ability to deeply understand customers' preferences and shopping intents. In this paper, we introduce the MCM (**M**ulti-task pre-trained **C**ustomer **M**odel), a large pre-trained BERT-based multi-task customer model with 10 million trainable parameters for e-commerce stores. This model aims to empower all personalization projects by providing commonly used preference scores for recommendations, customer embeddings for transfer learning, and a pre-trained model for fine-tuning. In this work, we improve the SOTA BERT4Rec framework to handle heterogeneous customer signals and multi-task training as well as innovate new data augmentation method that is suitable for recommendation task. Experimental results show that MCM outperforms the original BERT4Rec by 17% on on NDCG@10 of next action prediction tasks. Additionally, we demonstrate that the model can be easily fine-tuned to assist a specific recommendation task. For instance, after fine-tuning MCM for an incentive based recommendation project, performance improves by 60% on the conversion prediction task and 25% on the click-through prediction task compared to a baseline tree-based GBDT model.

CCS Concepts: • **Information systems** → **Recommender systems**.

Additional Key Words and Phrases: personalization, transformer, multi-task, recommendation, pre-train

ACM Reference Format:
Rui Luo, Tianxin Wang, Jingyuan Deng, and Peng Wan. 2018. MCM: A Multi-task Pre-trained Customer Model for Personalization. *Proc. ACM Meas. Anal. Comput. Syst.* 37, 4, Article 111 (August 2018), 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

In a personalized recommendation system, it is critical to understand each customer's preference and shopping intent holistically based on customers' profile as well as comprehensive historical behaviors like browsing, searching and purchasing signals. E-commerce stores usually have tens of billions of customer historical behavior data. These signals, if learned with a large capacity model, can help to provide the grounding of customers understanding and be utilized by thousands of downstream use cases. Bert-based pre-trained model like GPT has been successful in NLP and CV [4, 5, 8], Researchers in recommendation field explore Bert-based model on specific task like session-based recommendation [9], under the setting of sequential recommendation [2, 3, 6, 7]. However, large-scale pre-training with Bert is still largely under-explored in the field of recommender systems.

*Both authors contributed equally to this research.

Authors' addresses: Rui Luo, luorui@amazon.com; Tianxin Wang, watianxj@amazon.com, Amazon LLC; Jingyuan Deng, Amazon LLC, jingyua@amazon.com; Peng Wan, Amazon LLC, wanp@amazon.com.

Manuscript submitted to ACM

In this paper, we propose **M**ulti-task pre-trained **C**ustomer **M**odel (MCM) model: a large-capacity Bert-based multi-task model with 10 million trainable parameters, pre-trained on a vast amount of behavior data from a large e-commence service. We make architectural improvements for better pre-training, which will be explained in more detail in the model section. Offline results show that MCM outperform the original Bert4rec on multi-tasks by average 17% on NDCG@10 of next action prediction. In order to evaluate the ability to support new personalization tasks, we fine-tune the model for an incentive offer recommendation task, the performance improves by 60% on conversion rate and 25% on click-through rate, compared to the baseline GBDT model.

2 METHODOLOGY

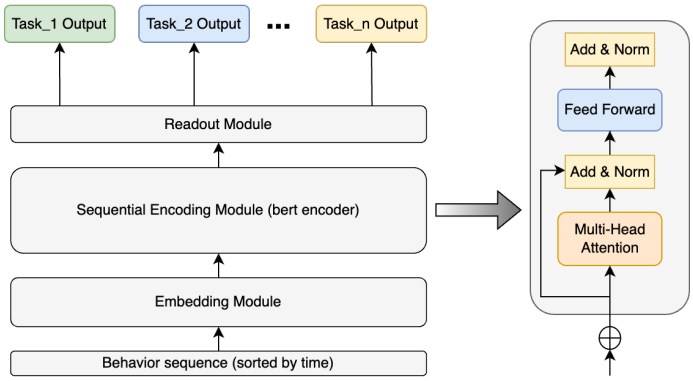

Fig. 1. MCM model structure.

2.1 Model Framework

MCM consists of three modules: embedding module, sequential encoding module and readout module1. We inherited the sequential encoding module from Bert4rec, while make algorithm improvements on the other two modules, which we will introduce in more detail in the following sessions.

2.1.1 Heterogeneous Embedding Module. In this module, we convert the raw inputs into distributed representations through embedding lookup, as is typically done in bert-based models.

The original input is a heterogeneous interaction sequence including purchase actions and non-purchase actions. Non-purchase actions are actions that are High Value Actions (HVAs) customers have with the e-commerce stores, e.g. member sign-up, mobile camera search, click records of products etc. Instead of feeding the raw inputs into the embedding module, we choose to represent each interaction i with a set of features f_i^j, $j \in 1, 2, ..., |J|$, where $|J|$ denotes the total number of features for each interaction. Now the inputs of each customer c are $|J|$ sequences, with the j-th sequence in the form of $F^j = [f_1^{(j)}, f_2^{(j)}, ..., f_{n_i}^{(j)}]$. Currently, the features include the hierarchical structures of the product: product line, category, subcategory as well as brand. Additionally, we design a feature called token type to handle heterogeneous input, making it easier for the model to differentiate different types of interactions.

Each distinct feature value is assigned a unique embedding vector. After the embedding look-up, the inputs are converted to $|J|$ sequences of embeddings, we perform average pooling to these sequences, producing a single embedding sequence $E = [e_1, e_2, ..., e_{n_i}]$.

2.1.2 Task-Aware Attentional Readout Module. The output of the sequential encoding module is a sequence of hidden vectors, with the same length as the input sequence. Previous work[9] computes the inner product between the last hidden vector and the item embedding to produce the score for the corresponding item. This can be sub-optimal since the last hidden vector is a fixed representation of the whole behavior sequence, which is not aware of the specific item or task to predict. Different tasks may be related to different behaviors within the whole behavior sequence.

We propose a novel task-aware attentional readout module, which allows different items (labels in each task) and different tasks to attend to different subsequences of the hidden sequence with attention mechanism, in order to produce a task-specific representation. Specifically, let h_i denote the i-th embedding of the output of the sequential encoding module, and let e_w denote the embedding for a particular item w in a certain task, the attentional readout operation can be described as:

$$a_{w,i} = \frac{\exp((e_w)^\top h_i)}{\sum_i \exp((e_w)^\top h_i)}, \tag{1}$$

$$r_w = \sum_i a_{w,i} h_i, \tag{2}$$

where r_w is the representation of the input sequence, specific to item w. The predicted score for item w is: $s(w) = r_w^\top e_w$. Softmax operation is performed on the scores to produce the final distribution.

2.2 Model Learning with prefix-augmentation

Previous work[9] on sequential recommendation adopt a popular augmentation method in NLP called masked language model, which randomly masks out some tokens in the input sequence and asks the model to predict them based on all other tokens. Such augmentation is suitable for language modeling, but we believe it can be problematic for recommendation tasks since it leaks future information. We propose a new augmentation method called random prefix augmentation, which randomly samples a prefix from the whole input sequence, and ask the model to predict the last item. In this case, the input will only include the items before the last item, so that our augmentation avoids leaking future information. For example, suppose the original input sequence is $[i_1, i_2, i_3]$, valid prefixes include $[i_1]$ and $[i_1, i_2]$. The augmentation is performed at batch time rather than during data pre-processing, in order to save memory.

The loss function for each prefix is defined as the negative log-likelihood of the label item (the last item): L = - $\log P(i = i_{gt}|S), where i_{gt}$ denotes the ground truth item, S denotes the input sequence, which contains all items but the last one. For multi-task training, the loss of all tasks are summed together.

3 EXPERIMENT

3.1 Experiment Setup

3.1.1 Datasets and Evaluation Details. To train our models, we use data from a large e-commerce service. We use customer behaviour data sampled from 6 years customer history. The dataset consists of 40M customers and 10B interactions. The behavior sequence includes three types of interactions: item purchases, item clicks and customer valuable actions. For each purchase and click interaction, we use products' hierarchical features including product line (PL), category, subcategory as well as brands. To note, these features are also the tasks we train the model to predict on customer's next preferences. As suggested in [1], we split the dataset into training, validation and testing dataset by time to avoid leaking future information. We adopt ranking metrics for evaluation, the primary metric is **NDCG**

Table 1. Performance of preference scores.

*The metrics in this table are calculated for the top 10 positions

	PL		Brand		Category		Subcategory	
Model	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall
Bert4rec	0.6532	0.8713	0.0104	0.0155	0.0489	0.0721	0.0411	0.0602
MCM-*Single*	0.7039	0.9129	0.0127	0.0186	0.0532	0.0768	0.0454	0.0648
MCM-*MTL*	0.716	0.9155	0.0128	0.0185	0.0559	0.0772	0.0457	0.0651
MCM-*Final*	0.725	0.9203	0.0132	0.019	0.0562	0.0775	0.0465	0.0653

(Normalized Discounted Cumulative Gain), as well as recall and precision. The Bert encoder consists of three transfomer layers, the head number is four. The maximum sequence length is truncated to 300.

3.2 Quality of Preference Scores

We first compare the performance between our model and a SOTA sequential recommendation model Bert4rec. As illustrated in Table.1, MCM_final significantly outperforms bert4rec by about 11%. We also conduct ablation experiments with variants of MCM model MCM_Single and MCM_MTL. Compared to Bert4rec, MCM_Single utilizs heterogeneous interaction sequence and contextual features to train each single task, and MCM_MTL utilizs attentional readout and MTL. MCM-Final utilizes heterogeneous data, attentional readout and pre-fix data augmentation. The results show that richer input signals contribute to the incrementality the most, while MTL and prefix augmentation are also helpful.

3.3 Extensibility of MCM

To demonstrate the flexibility and extensibility of MCM, we give a detailed example showing how to modify and fine-tune the model on a next action recommendation use case. This task aims to encourage customers to complete one action task by providing incentives (e.g. cash backs). There are in total 28 tasks, including purchasing from a new product line (i.e., the product line that the customer never bought before) and trying new services like camera search or prime video streaming service.

The multi-task prediction scores from MCM model have covered the tasks and can be directly utilized, however it may not reflect customers' behaviors with incentives. So we add a new head on top of the sequential encoder to predict the incentive effect and fine-tune the model with customer behavior data under incentives: 30K action clicks and completion(conversion) records. From the results, MCM significantly outperforms a tree-based GBDT task prediction model by 25% on conversion NDCG, and fine-tuned model (MCM_finetuned) outperforms the MCM model without fine-tuning by 35% on conversion NDCG, which in total drives over 60% improvement on conversion rate as compared to tree-based GBDT model.

4 CONCLUSION

In this paper, we introduce MCM, a large pre-trained customer model that serves as a sequential multi-task recommendation model to support diverse personalization projects. Through experiments, we demonstrate the model's ability to provide highly accurate preference predictions, which surpass the performance of other baseline models. We also showcase a detailed use case on a recommendation project, demonstrating how MCM can be extended to new tasks and deliver significant performance improvements.

5 SPEAKER BIO

Rui Luo and Tianxin Wang are applied scientists in Amazon, they work on recommender systems to improve customer shopping experience.

REFERENCES

[1] P. Covington, J. Adams, and E. Sargin. Deep neural networks for youtube recommendations. In *RecSys*. ACM, 2016.

[2] B. Hidasi and A. Karatzoglou. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 843–852, 2018.

[3] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015.

[4] J. Lu, D. Batra, D. Parikh, and S. Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

[5] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[6] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 811–820, 2010.

[7] G. Shani, D. Heckerman, R. I. Brafman, and C. Boutilier. An mdp-based recommender system. *Journal of Machine Learning Research*, 6(9), 2005.

[8] S. Shen, L. H. Li, H. Tan, M. Bansal, A. Rohrbach, K.-W. Chang, Z. Yao, and K. Keutzer. How much can clip benefit vision-and-language tasks? In *International Conference on Learning Representations*.

[9] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450, 2019.