A Transformer-based Embedding Model for Personalized Product Search

Keping Bi
University of Massachusetts Amherst
kbi@cs.umass.edu

Qingyao Ai
University of Utah
aiqy@cs.utah.edu

W. Bruce Croft
University of Massachusetts Amherst
croft@cs.umass.edu

ABSTRACT

Product search is an important way for people to browse and purchase items on E-commerce platforms. While customers tend to make choices based on their personal tastes and preferences, analysis of commercial product search logs has shown that personalization does not always improve product search quality. Most existing product search techniques, however, conduct undifferentiated personalization across search sessions. They either use a fixed coefficient to control the influence of personalization or let personalization take effect all the time with an attention mechanism. The only notable exception is the recently proposed zero-attention model (ZAM) that can adaptively adjust the effect of personalization by allowing the query to attend to a zero vector. Nonetheless, in ZAM, personalization can act at most as equally important as the query and the representations of items are static across the collection regardless of the items co-occurring in the user's historical purchases. Aware of these limitations, we propose a transformer-based embedding model (TEM) for personalized product search, which could dynamically control the influence of personalization by encoding the sequence of query and user's purchase history with a transformer architecture. Personalization could have a dominant impact when necessary and interactions between items can be taken into consideration when computing attention weights. Experimental results show that TEM outperforms state-of-the-art personalization product retrieval models significantly.

KEYWORDS

Product Search; Personalization; Transformer

ACM Reference Format:
Keping Bi, Qingyao Ai, and W. Bruce Croft. 2020. A Transformer-based Embedding Model for Personalized Product Search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3397271.3401192

1 INTRODUCTION

Product search systems have been playing an important role in serving customers shopping on online e-commerce platforms in their daily life. Usually, people issue queries about their shopping needs on the platform and purchase items from the search results based on their personal tastes and preferences. Aware of this point, recent studies have explored to incorporate personalization in product search and achieved compelling results [1, 2, 8].

Despite its great potential, personalization does not always improve the quality of product search. Based on the analysis of commercial search logs, Ai et al. [1] have observed that personalized models can outperform non-personalized models only on the queries where the preferences of individuals significantly differ from the group preference. While applying a universal personalization mechanism sometimes could be beneficial by providing more information about user preferences, especially when the query carries limited information, unreliable personal information could also harm the search quality due to data sparsity and the introduction of unnecessary noise. Therefore, it is essential to determine when and how to conduct personalization under various scenarios.

Most existing personalized product search models, however, do not conduct differential personalization adaptively under different contexts. Ai et al. [2] propose to control the influence of personalization by representing the users' purchase intent with a convex combination between the query embedding and user embedding. This method applies undifferentiated personalization to all search sessions since the coefficient of the combination is a fixed number. Guo et al. [8] fuse query and users' long and short-term preferences to indicate the users' specific intention. While the long and short-term preferences are modeled by attending to the users' recent purchases and a global user vector with the query, the model itself still conducts personalization all the time. Later, Ai et al. [1] proposed a zero attention model (ZAM) which introduces a zero vector that the query can attend to besides users' previous purchases. In contrast to [8], by allowing the zero vector to have attention weights, the influence of personalization can be controlled. Nonetheless, despite the ability to adaptively personalize a query-user pair, the maximum personalization ZAM can perform is to equally consider the query and the user information, which may be not enough when the user preference dominates the purchase.

In this paper, we propose a transformer-based embedding model (TEM) that is more flexible where personalization can vary from no to full effect. As we will demonstrate later in Section 3, a single-layer TEM is similar to ZAM but with a larger range of controlling personalization. A multiple-layer TEM takes into consideration the interactions between purchased items so that it could learn potentially better dynamic representations of queries and items, which probably lead to better attention weights. We also compare and analyze the ability of personalization between our model and the zero-attention model theoretically in Section 3.5. Our experimental results on the Amazon product search dataset [10, 12] show that TEM significantly outperforms state-of-the-art baselines.

SIGIR '20, July 25–30, 2020, Virtual Event, China
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401192

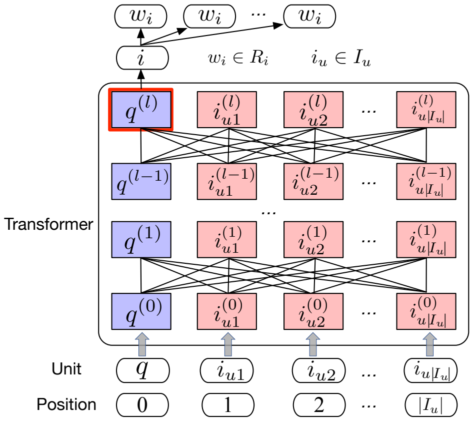

Figure 1: Our Transformer-based Embedding Model (TEM).

2 RELATED WORK

Product Search. Earlier work on product search mainly considers products as structured entities and uses facets for the task [13]. Language model based approaches have been studied [7] for keyword search. To alleviate word mismatch problems, more recently, Van Gysel et al. [12] introduce a latent semantic entity model that matches products and queries in the latent semantic space. Learning to rank techniques have also been investigated [9]. In the scope of personalized product search, Ai et al. [2] use a convex combination between query and user embeddings for personalization; Guo et al. [8] represent users' long and short-term preferences with an attention mechanism; Ai et al. [1] provide insight on when personalization could be beneficial and propose a zero-attention model to control how personalization takes effect. Personalization has also been studied in multi-page product search [4].

Transformer-based Retrieval Models. Studies on retrieval with transformers have been sparse and most of them leverage pretrained contextual language models, i.e., BERT [6], which is grounded on the transformer architecture. It achieves compelling performance on a wide variety of tasks such as passage ranking [11] and document retrieval [5].

3 TRANSFORMER-BASED EMBEDDING MODEL (TEM)

In this section, we first introduce each component of TEM as shown in Figure 1 and then compare TEM with ZAM theoretically.

3.1 Item Generation Model

We use an item generation model to capture the purchase relationship between an item and query-user pairs. This embedding-based generation framework has been shown to be effective by previous studies on personalized product search [1–3]. Formally, let q be a query issued by a user u and i be an item in S_i which is the set of all the items in a collection. The probability of i being purchased by u given q is modeled as

$$P(i|q,u) = \frac{\exp(\mathbf{i} \cdot \mathbf{M}_{qu})}{\sum_{i' \in S_i} \exp(\mathbf{i'} \cdot \mathbf{M_{qu}})} \tag{1}$$

where $\mathbf{i} \in \mathbb{R}^d$ is the vector representation of dimension size d and $\mathbf{M_{qu}}$ is the representation by jointly modeling the query-user pair (q,u). We will elaborate how to yield $\mathbf{M_{qu}}$ later.

3.2 Query Representation

As shown in previous studies [1, 2, 12], an effective way to encode query is to apply a non-linear projection ϕ on the average query word embeddings:

$$\mathbf{q} = \phi(\{w_q|w_q \in q\}) = \tanh(W_\phi \cdot \frac{\sum_{w_q \in q} \mathbf{w_q}}{|q|} + b_\phi) \tag{2}$$

where $W_\phi \in \mathbb{R}^{d \times d}$ and $b_\phi \in \mathbb{R}^d$, $|q|$ is the length of query q, and $\mathbf{w_q} \in \mathbb{R}^d$ is the embedding of word w_q in q. This way of encoding queries has outperformed other techniques such as using average word embeddings and applying recurrent neural networks on the word embedding sequence for product search [2].

3.3 Item Language Model

As in [2, 3], item embeddings are learned from their associated reviews. Let R_i be the set of words in the reviews associated with item i. Embeddings of words and items are optimized to maximize the likelihood of observing R_i given i:

$$P(R_i|i) = \prod_{w \in R_i} \frac{\exp(\mathbf{w} \cdot \mathbf{i})}{\sum_{w' \in V} \exp(\mathbf{w'} \cdot \mathbf{i})} \tag{3}$$

where V is the vocabulary of words in the corpus.

3.4 Transformer-based Personalization

Different queries may need various degrees of personalization [1]. Some can be satisfied with popular items in general and some correlates closely with users' historical purchases. To represent the purchase intent with query-dependent personalization, we leverage a transformer encoder [14] architecture to capture the interaction between query and users' historical purchased items, as shown in Figure 1. Let $I_u = (i_{u1}, i_{u2}, \cdots, i_{u|I_u|})$ be the sequence of items purchased by u in a chronological order, the size of which is $|I_u|$. We feed the sequence (q, I_u) as the input to a l-layer transformer encoder. Since a recent purchase may play a different role compared with a long-ago purchase, in addition to query and item embeddings of a corresponding unit (query or item), positional embeddings (*PosEmb*) are used to indicate the purchase order of each item. The input vectors to the transformer are:

$$\mathbf{q}^{(0)} = \mathbf{q} + PosEmb(0);\ \mathbf{i^{(0)}_{uk}} = \mathbf{i_{uk}} + PosEmb(k), i_{uk} \in I_u \tag{4}$$

where $\mathbf{q}$ and $\mathbf{i_{uk}}$ can be computed according to Eq. 2 & 3 respectively.

Then user u's purchase intent given q, i.e., M_{qu}, can be represented with the output vector of query q at the l-th layer, i.e.,

$$\mathbf{M_{qu}} = \mathbf{q^{(l)}} \tag{5}$$

We use $\mathbf{q^{(l)}}$ as M_{qu} because it is computed by attending to each transformer input using query q which is more reasonable than other output vectors that attend to the input with a previously purchased item. Specifically, $\mathbf{q^{(l)}}$ is computed as a weighted combination of embeddings of query and purchased items from the

Table 1: Statistics of the Amazon datasets.

Dataset	Cell Phones	Sports	Movies
#Users	27,879	35,598	123,960
#Items	10,429	18,357	50,052
#Reivews	194,439	296,337	1,697,524
#Queries	165	1,543	248

previous transformer layer followed by a projection function:

$$\mathbf{q}^{(\mathbf{l})}=g\Bigg(\frac{\exp\Big(f(q_Q^{(l-1)},q_K^{(l-1)})\Big)}{\exp\Big(f(q_Q^{(l-1)},q_K^{(l-1)})\Big)+\sum_{i'\in I_u}\exp\Big(f(q_Q^{(l-1)},i_K'^{(l-1)})\Big)}\cdot q_V^{(l-1)} \\ +\sum_{i\in I_u}\frac{\exp\Big(f(q_Q^{(l-1)},i_K^{(l-1)})\Big)}{\exp\Big(f(q_Q^{(l-1)},q_K^{(l-1)})\Big)+\sum_{i'\in I_u}\exp\Big(f(q_Q^{(l-1)},i_K'^{(l-1)})\Big)}\cdot i_V^{(l-1)}\Bigg) \tag{6}$$

In Eq. 6, $f(x,y)$ computes attention score of y with respect to x. As in [14], g is a projection function that firstly applies f for multiple attention heads followed by a feed-forward layer and residual connections for both the multi-head attention sub-layer and the feed-forward sub-layer. $q_Q^{(l-1)}$, $q_K^{(l-1)}$, and $q_V^{(l-1)}$ are computed according to:

$$q_Q^{(l-1)} = \mathbf{q}^{(l-1)}W_q^Q; q_K^{(l-1)} = \mathbf{q}^{(l-1)}W_q^K; q_V^{(l-1)} = \mathbf{q}^{(l-1)}W_q^V \tag{7}$$

where $W_q^Q \in \mathbb{R}^{d\times(d/h)}$, $W_q^K \in \mathbb{R}^{d\times(d/h)}$ and $W_q^V \in \mathbb{R}^{d\times(d/h)}$ are projection matrices; $\mathbf{q}^{(\mathbf{l-1})}$ is the embedding of q at the $(l-1)$-th transformer layer; and h is the number of attention heads. $i_K^{(l-1)}$ and $i_V^{(l-1)}$ in Eq. 6 are computed similarly based on $\mathbf{i}^{(\mathbf{l-1})}$, i.e., the vector of i at $(l-1)$-th layer. In this way, TEM can have the capability of ZAM to coordinate personalization and is more general and flexible, as shown in the next section where we will illustrate the relation between TEM with ZAM.

3.5 Comparison with Zero-attention Model

In ZAM [1], query and user are jointly modeled by:

$$\mathbf{M_{qu}} = \mathbf{q} + \sum_{i\in I_u} \frac{\exp\left(f'(q,i)\right)}{\exp\left(f'(q,\mathbf{0})\right) + \sum_{i'\in I_u}\exp\left(f'(q,i')\right)} \cdot \mathbf{i} \tag{8}$$

where f' is a multi-head attention function. In ZAM, when q does not require personalization or it has no useful purchase history to attend to, all the items in I_u would have small attention weights, which allows M_{qu} to include information only from q. When personalization has great potential for q, most attention is allocated to the historical purchases I_u rather than the zero vector. Eq. 8 shows that the maximum personalization ZAM can conduct is to consider I_u equally important to q. However, in some cases, personalization could have a larger impact than queries. Ai et al. [2] has shown that the optimal query weight could be much lower than the user weight on some product categories where personalization is indispensable. In contrast, TEM based on Eq. 6 can learn to balance the influence of personalization for each query automatically without limits on the personalization degree. Specifically, the query weight can be as small as 0 when personalization is dominant and as large as 1 when personalization is not needed at all.

In addition, when $l = 1$, $\mathbf{q}^{(\mathbf{l-1})}$ and $\mathbf{i}^{(\mathbf{l-1})}$ in Eq. 6 become $\mathbf{q}^{\mathbf{0}}$ and $\mathbf{i}^{(\mathbf{0})}$ (shown in Eq. 4) respectively. In this case, the only difference between query and items representations of TEM and ZAM is the positional embeddings. When $l > 1$, $\mathbf{q}^{(\mathbf{l-1})}$ and $\mathbf{i}^{(\mathbf{l-1})}$ are learned from previous transformer layers by interacting with all the units in the sequence (q, I_u). In this way, the query and items are dynamically represented depending on its interaction with the other units associated with this q-u pair rather than having static vectors across the corpus. By considering the relation between historical purchased items, e.g., same brands or categories, TEM could learn potentially better representation to facilitate product search.

4 EXPERIMENTS

Datasets. We use the Amazon product search dataset [10] for experiments, as in previous work [2, 3, 12]. Since there are no available queries for this dataset, we construct queries for each item following the same strategy as in [2, 3, 12]. A query string of each purchased item is formed by concatenating words in the multi-level category of the item and removing stopwords as well as duplicate words. In this way, there could be multiple queries for each item since an item may belong to multiple categories. The user and each query associated with her purchased item are considered as the possible query-user pairs that lead to purchasing the item. We use three categories of different scales for experiments, which are *Cellphones&Accessories*, *Sports&Outdoors* and *Movies&TV*. The statistics are shown in Table 1.

Evaluation. We randomly divide 70% of all the available queries into the training set and the rest 30% queries are shared by validation and test sets. If all the queries of a purchased item fall in the test set, we randomly put one query back to the training set. We partition the purchases of a user to the training/validation/test set according to the ratio 0.8/0.1/0.1 in a chronological order. For any purchase in the validation or test set, if none of the queries associated with the purchased item are in the query set for validation and test, this purchase will be moved back to the training set. Our partition ensures that the purchases in the test set happen after the purchases in the training set and no test query has been seen in the training set. We use MRR, Precision, and NDCG at 20 as the metrics.

Baselines. We include five representative product search models as baselines: the Latent Semantic Entity model (LSE) [12] which is an embedding-based non-personalized model; Query Embedding Model (QEM) [1], another non-personalized model, which conducts item generation (Sec. 3.1 & 3.2) based on the query alone; Hierarchical Embedding Model (HEM) [2] which balances the effect of personalization by applying a convex combination of user and query representation; the Attention-based Embedding Model (AEM) [1] which constructs query-dependent user embeddings by attending to users' historical purchases with query, similar to the attention model proposed by Guo et al. [8]; a state-of-the-art model: the Zero Attention Model (ZAM) [1] which introduces a zero vector to AEM so that the influence of personalization can be differentiated for various queries. We only include neural models as our baselines since term-based models have been shown to be much less effective for product search in previous studies [1–3].

Training. We train our model and all the baselines for 20 epochs with 384 samples in each batch. We set the embedding size of all the models to 128 and sweep the number of attention heads h from {1,2,4,8} for attention-based models. The number of transformer

Table 2: Comparison between the baselines and our proposed TEM. '*' marks the bast baseline performance. '†' indicates significant improvements over all the baselines in paired student t-test with $p < 0.05$.

Dataset		Cell Phones & Accessories			Sports & Outdoors			Movies & TV		
Model		*MRR*	*NDCG*@20	*P*@20	*MRR*	*NDCG*@20	*P*@20	*MRR*	*NDCG*@20	*P*@20
Non-personalized	LSE	0.013	0.022	0.004	0.010	0.021	0.004	0.010	0.015	0.002
	QEM	0.029	0.036	0.003	0.031	0.044	0.006	0.004	0.006	0.001
Personalized	HEM	0.044^*	0.057^*	0.006^*	0.032	0.049	0.007^*	0.007	0.011	0.002
	AEM	0.043	0.049	0.004	0.031	0.045	0.006	0.013^*	0.020	0.003^*
	ZAM	0.041	0.046	0.004	0.040^*	0.057^*	0.007^*	0.013^*	0.022^*	0.003^*
	TEM	$\mathbf{0.056}^\dagger$	$\mathbf{0.072}^\dagger$	$\mathbf{0.007}^\dagger$	$\mathbf{0.049}^\dagger$	$\mathbf{0.074}^\dagger$	$\mathbf{0.010}^\dagger$	$\mathbf{0.020}^\dagger$	$\mathbf{0.028}^\dagger$	$\mathbf{0.004}^\dagger$

layers l is chosen from {1,2,3} and the dimension size of the feed-forward sub-layer of the transformer is set from {96, 128, 256, 512}. Adam with learning rate 0.0005 is used to optimize the models.

Results. Table 2 shows the ranking performance of the baseline models and TEM [1]. Similar to previous studies [1–3], we observe that LSE and QEM perform worse than personalized product search baselines in most cases. If we compare the personalized product search baselines, HEM has the best performance on *Cell Phones* whereas ZAM performs the best on *Sports* and *Movies*. Specifically, HEM and AEM achieve better results than ZAM on *Cell Phones* and worse results on the other two datasets. This indicates that, while adjusting the influence of personalization with the attention weights on the zero vector could benefit the retrieval performance of ZAM, its limitation on personalization (i.e., the personalization weight can be no larger than the query weight) could harm the search quality on datasets where personalization is essential.

On all the categories, TEM achieves the best performance in terms of all the three metrics. The improvement upon the best baseline on each dataset is approximately 20% to 50%. From the improvement of Precision, NDCG, and MRR, we can infer that TEM not only retrieves more ideal items in the top 20 results but also promotes them to higher positions. This demonstrates that TEM can benefit the effectiveness of personalized models with a more flexible mechanism to control the influence of personalization and by learning dynamic item representations with the interaction between items taken into consideration.

Effect of Layer Numbers. We varied the number of transformer layers to see whether a single-layer or multi-layer transformer will lead to better results on each dataset. The best performance of TEM is achieved when l in Eq. 6 is set to 2 on *Sports* and 1 on *Cell Phones* as well as *Movies*. This indicates that considering the interactions between items does benefit the personalized product search models in some product categories.

5 CONCLUSION AND FUTURE WORK

In this paper, we propose a transformer-based embedding model, abbreviated as TEM, that can conduct query-dependent personalization. By encoding the sequence of the query and users' purchase history with a transformer architecture, the effect of personalization can vary from none to domination. We theoretically compare TEM with ZAM [1] and show that a single-layer TEM is an advanced version of ZAM with more flexibility and a multi-layer TEM extends the model with stronger learning abilities by incorporating the interactions between items co-occurring in users' purchase history. Our experiments empirically demonstrate the effectiveness of TEM by showing that TEM outperforms the state-of-the-art personalized product search baselines significantly. For future work, we consider studying TEM for explainable product search. The attention scores in TEM indicate the personalization degree and which historical items draw more attention for retrieving a result. This information could be helpful for users to make purchase decisions. In addition, we are also interested in incorporating other information about products such as price, ratings, and images with a transformer architecture to facilitate personalized product search.

ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

REFERENCES

[1] Qingyao Ai, Daniel N Hill, SVN Vishwanathan, and W Bruce Croft. 2019. A zero attention model for personalized product search. In *CIKM'19*. 379–388.

[2] Qingyao Ai, Yongfeng Zhang, Keping Bi, Xu Chen, and W Bruce Croft. 2017. Learning a hierarchical embedding model for personalized product search. In *SIGIR'17*. ACM, 645–654.

[3] Keping Bi, Qingyao Ai, Yongfeng Zhang, and W Bruce Croft. 2019. Conversational product search based on negative feedback. In *CIKM'19*. 359–368.

[4] Keping Bi, Choon Hui Teo, Yesh Dattatreya, Vijai Mohan, and W Bruce Croft. 2019. A Study of Context Dependencies in Multi-page Product Search. In *CIKM'19*.

[5] Zhuyun Dai and Jamie Callan. 2019. Deeper text understanding for IR with contextual neural language modeling. In *SIGIR'19*. 985–988.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Huizhong Duan, ChengXiang Zhai, Jinxing Cheng, and Abhishek Gattani. 2013. A probabilistic mixture model for mining and analyzing product search log. In *CIKM'13*. ACM, 2179–2188.

[8] Yangyang Guo, Zhiyong Cheng, Liqiang Nie, Yinglong Wang, Jun Ma, and Mohan Kankanhalli. 2019. Attentive long short-term preference modeling for personalized product search. *TOIS* 37, 2 (2019), 1–27.

[9] Shubhra Kanti Karmaker Santu, Parikshit Sondhi, and ChengXiang Zhai. 2017. On application of learning to rank for e-commerce search. In *SIGIR'17*. ACM, 475–484.

[10] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *SIGKDD'15*. ACM, 785–794.

[11] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *arXiv preprint arXiv:1901.04085* (2019).

[12] Christophe Van Gysel, Maarten de Rijke, and Evangelos Kanoulas. 2016. Learning latent vector spaces for product search. In *CIKM'16*. ACM, 165–174.

[13] Damir Vandic, Flavius Frasincar, and Uzay Kaymak. 2013. Facet selection algorithms for web product search. In *CIKM'13*. ACM, 2327–2332.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[1] The numbers in Table 2 are smaller than those reported by Ai et al. [2] since they randomly split user purchases to training and test set which makes the prediction of purchases in their test set easier than predicting future purchases in our test set.