PersonalTM: Transformer Memory for Personalized Retrieval

Ruixue Lian*
Department of Electrical and Computer Engineering,
University of Wisconsin-Madison
Madison, WI, USA
ruixue.lian@wisc.edu

Sixing Lu, Clint Solomon, Gustavo Aguilar, Pragaash Ponnusamy, Jialong Han, Chengyuan Ma, Chenlei Guo
Amazon Alexa AI
Seattle, WA, USA
cynthialu,matclint,gustalas,ponnup
jialongh,mchengyu,guochenl@amazon.com

ABSTRACT

The Transformer Memory as a Differentiable Search Index (DSI) [32] has been proposed as a new information retrieval paradigm, which aims to address the limitations of dual-encoder retrieval framework based on the similarity score. The DSI framework outperforms strong baselines by directly generating relevant document identifiers from queries without relying on an explicit index. The memorization power of DSI framework makes it suitable for personalized retrieval tasks. Therefore, we propose a Personal Transformer Memory (PersonalTM) architecture for personalized text retrieval. PersonalTM incorporates user-specific profiles and contextual user click behaviors, and introduces hierarchical loss in the decoding process to align with the hierarchical assignment of document identifier. Additionally, PersonalTM also employs an adapter architecture to improve the scalability for index updates and reduce computation costs, compared to the vanilla DSI. Experiments show that PersonalTM outperforms the DSI baseline, BM25, fine-tuned dual-encoder, and other personalized models in terms of precision at top 1st and 10th positions and Mean Reciprocal Rank (MRR). Specifically, PersonalTM improves p@1 by 58%, 49%, and 12% compared to BM25, Dual-encoder, and DSI, respectively.

CCS CONCEPTS

• **Information systems** → **Personalization**.

KEYWORDS

Search, Personalization, Transformer Memory

ACM Reference Format:
Ruixue Lian and Sixing Lu, Clint Solomon, Gustavo Aguilar, Pragaash Ponnusamy, Jialong Han, Chengyuan Ma, Chenlei Guo. 2023. PersonalTM: Transformer Memory for Personalized Retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, Austin, TX, USA, 5 pages. https://doi.org/10.1145/3539618.3592037

*Work was done as an intern at Amazon Alexa AI.

SIGIR '23, July 23–27, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07…$15.00
https://doi.org/10.1145/3539618.3592037

1 INTRODUCTION

In recent years, IR has undergone substantial progress owning to the resurgence of deep neural networks, with transformer-based Language Models (LMs) playing a particularly significant role. These models possess the capability of learning natural language representations from vast amounts of data, contributing to the advancement of IR [3, 11, 14, 20, 29, 30, 33]. Considering the Dual Encoder (DE) approach [9, 39] is unable to learn the deep interaction between queries and documents, several sequence-to-sequence frameworks have been proposed for IR, enabling generate documents relevant to the queries directly [2, 8, 18, 19, 21, 31, 32, 35, 42]. For instance, WebGPT proposes a method to answer long-form questions in a text-based web-browsing environment by fine-tuning GPT-3 [3, 28], DSI utilizes a transformer-based encoder-decoder network to generate a ranked list of relevant document id [32]. The goal of DSI is to implicitly learn the relationship between query document id, and between document and document id, by encoding all information into the model parameters. This framework simplifies the system architecture by utilizing a single LM, and leverages the memorization ability of the transformer.

Meanwhile, personalized retrieval is becoming increasingly important, aiming to refine queries and tailor retrieval results to specific preferences of individual users [10, 25, 26, 38, 41, 43]. Numerous studies have been made to personalized retrieval, demonstrating precision improvement by incorporating user profiles and contextual history [22, 24]. For example, P-Click [1] reranks the documents based on a user's clicks for a given query, and SLTB [12] outputs personalized ranking list by utilizing diverse clicked-based or topic-based features. [13, 40] uses RNN-based networks to extract short- and long-term user profiles from personalized historical information and apply it to DE.

In this work, we enhance performance for personalized retrieval by incorporating user identification, contextual history, and user click behaviors into the transformer memory framework. Our method involves a novel decoding architecture and applying hierarchical loss aligned with the document id generation. Furthermore, we propose a flexible adaptor strategy to facilitate retraining in the event of an index update. The main contributions of our work are:

- Compared with other personalized retrieval works, we utilize an end-to-end LM to generate relevant document id given the query directly.
- We leverage user identifiers and user-specific contextual data as personalized features into the DSI architecture. Different features are fed to different decoder cross-attention

layers to effectively integrate the contextual features without increasing trainable parameters.
- We apply a hierarchical loss function at the decoding stage to further boost the performance aligned with the semantic hierarchical document id assignment.
- We adopt the prefix adapter to learn the interaction between query and contextual data. Model parameters and training time are reduced by 10x and 2x compared to fine-tuning, making it suitable for practical deployment.

2 PROPOSED METHODS

In this work, we propose a novel approach for personalized retrieval using a personal end-to-end transformer memory (PersonalTM). In this section, we describe a model architecture that integrates the two personalized features into the PersonalTM, with a hierarchical loss at the decoding stage to enhance performance. Additionally, we implement relevant information selection to optimize the utilization of contextual features and incorporate a prefix-adaptor for more agile training.

2.1 PersonalTM model architecture integrating personalized features

Given a query, we utilize a transformer-based encoder-decoder network (such as T5) as the base structure to generate relevant document ids with two types of additional personalized information: 1) user profiles such as user identifiers; 2) personal context such as user browsing histories in cookies. The model is trained to learn the relationships between the query and document id, document and document id, as well as the query and the personalized information.

As shown in Figure 1, let q be the query, p be static user identifier, d be target document, and h be personal context feature. We concatenate p to q into one sequence $[p;q]$ so that p is attended to q by self-attention layers in the encoder, so the model can learn the connection between this specific user and his preferred d. p is a synthetic user identifier constructed for each user by randomly selecting and concatenating k tokens from BERT tokenizer dictionary [27], we use $k = 4$. h is user-clicked documents.

Instead of concatenating h to q, we design a decoding structure so that they interact in the embedding space as shown in Figure 2 for the following reasons: 1) it is not effective to concatenate h with q due to input length limitation; 2) the long h cannot fully interact with q with simple concatenation; 3) decoding stage serves more on learning the connection between q and d, and that is also where h contribute to. To better use h since relevant contextual information would be more useful, we propose two similarity measurement modules, which are introduced in section 2.3.

Let f be the encoder, we extract $f([p;q])$ and $f(h)$ as shown in Figure 2, and input them to the higher and lower decoder layers via cross-attention, separately. Each feature representation is a vector with dimension $n \times l \times d$, where n is the batch size, l is the max length of input sequence, d is the feature dimension. Specifically, $f(h)$ input to the first decoder layer and $f([p;q])$ input to the rest decoding layers. This allows h to be integrated without adding additional layers or increasing the trainable parameters of the model. Because information about the past vanishes in left-to-right language models [34], the information injected in the upper layers would have higher weights than that in the lower levels, reflecting the different importance of $[p;q]$ and h.

Multiple works have shown the effectiveness of the fusion of different types of features at decoder layers via cross-attention [4–7]. We also experiment with another decoding structure that involves a fusion layer to merge $f([p;q])$ and $f(h)$, and feed this fused information to each decoder cross-attention layer. Specifically, we extract $f([p;q])$ and $f(h)$, and concatenate these embeddings by $[f([p;q]);f(h)]$. An MLP layer consisting of two linear layers is used to project $[f([p;q]);f(h)]$ to the original dimension aligning the decoder input. This way, query and personalized information are fused, and are input into every decoder cross-attention layer.

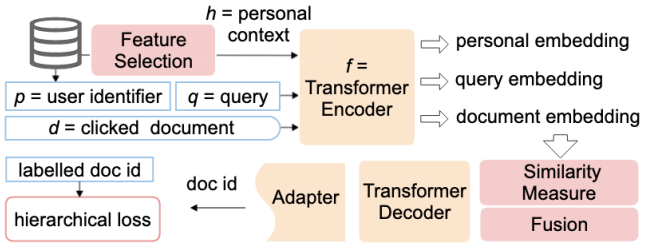

Figure 1: The training workflow of PersonalTM model architecture, integrating personalized features.

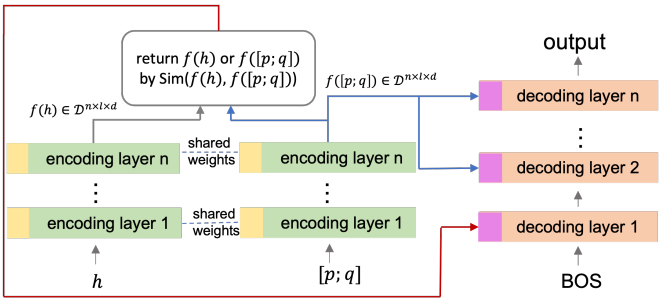

Figure 2: An overview of our proposed encoder-decoder architecture. A similarity measurement is used to select relevant contextual personalized information. $f([p;q])$ and $f(h)$ are fed into higher and lower decoder layers. Prefix-adaptor is applied and their parameters are marked in yellow and pink.

Transformer-based LMs have achieved state-of-the-art performance on many tasks. Still, their overall capacity or trainable parameters are prohibitively large for retraining, particularly for IR's frequent index updates. Previous work has proposed alternative lightweight model tuning approaches such as prefix tuning [16, 17, 23, 37]. In addition to the adaptor's superior performance on few-shot cases and its lower computation and storage requirements, we demonstrate that it is a promising method for personalization. By injecting additional continuous learnable parameters into each transformer layer, we achieve the updated model using the prefix adaptor. The training speed thus doubled since only the newly injected parameters are updated and the original LM is frozen during

training. The keys and values of the attention head in each self-attention and cross-attention layer are prepended with randomly initialized parameters. As shown in Figure 2, the newly injected prefix parameters are marked yellow and pink.

2.2 Hierarchical loss

Through semantic clustering, we assign each document a unique document id, similar to [32]. For instance, *any country music lyrics* and *country lyrics tabs chords for country music fan* should have closer or even the same high-level clusters because they have similar meanings. However, decoding the document id by digit has the problem of accumulated errors. To leverage the semantic hierarchy of the document id and penalize more severe semantic retrieval errors, we apply an additional hierarchical loss based on the document id hierarchy at the decoding steps. Therefore, prediction error on the first several digits in the document id (top-level clusters) will have a higher impact than lower digits (lower-level clusters) because a such error will end up with documents having different topics with the query.

The overall loss function we use is shown in E.q. (2), where the first term l_0 refers to cross-entropy loss as shown in E.q. (1)

$$l_0 = \text{cross-entropy(logits, labels)} \tag{1}$$

$$l = l_0 + w_i \cdot l_0; \ \sum_i w_i = 1, w_1 > w_2 > \ldots \geq w_{n-1} \geq w_n \tag{2}$$

The second term in E.q. (2) refers to the hierarchical loss. Specifically, we multiply each digit of the original loss with a scalar w_i, by which different penalties are applied to different positions in the document id, modulating their importance accordingly. We apply higher weights to higher-level positions as shown in E.q. (2).

2.3 Relevance denoise for personal context

Intuitively, only relevant personal context is useful because it implicitly increases the weights on tokens that imply user's preference. In contrast, irrelevant context introduces noise and thus misleads the model to the wrong predictions. For instance, given an incoming query *python running environment*. Among this user's browsed history, some of them (e.g. *python programming*) are relevant, while some (e.g. *what do pythons eat* or *country music*) are irrelevant to this query. Our experiment has demonstrated purely relying on a model to do denoise is very inefficient. Therefore, we adopt the following two mechanisms to retain as much useful personalized information for the predictions.

The first mechanism is feature selection for h on the sequence level among this user's browsing cookies, as shown in Figure 1. We apply a lightweight algorithm to select relevant personal context among user-browsed history. The algorithm selects the relevant documents according to the ratio of how many tokens in the document overlapped in the incoming query. The user-clicked documents in the session with this ratio greater than a certain value are considered personal context kept. We use the clicked documents in the latest session as the default context.

The second mechanism is similarity measurement between $f([p;q])$ and $f(h)$ in the latent space, as shown in Figure 2. A cosine similarity score s is calculated between $f([p;q])$ and $f(h)$. If s is greater than a certain threshold, h is kept and integrated into the decoding stage. Otherwise, h is omitted.

Algorithm 1 Relevant personal context selection

```
for row in testset do
    u_history = history[row["user_id"]], query = row["query"]
    for history in u_history do
        ratio = Similarity(history, query)
        relevant_history = list()
        if ratio >= threshold then
            relevant_history.append((ratio, history))
        end if
    end for
    return relevant_history[argmax(ratio)] if relevant_history
is not None else return u_history[t-1]
end for
```

3 EXPERIMENTS

3.1 Datasets

We use AOL4PS [15] dataset in our experiments. It contains the query, the corresponding clicked documents, and timestamps indicating the timeline of each user's search history. This dataset has a range of 12 weeks. We select relevant personal context from the first 9 weeks which is considered as historical data. The samples in the last 3 weeks are divided into training and test sets. The size of the training set and test set are 218,559 and 53,357, respectively. The number of distinct queries is 382,222 in the total dataset. Among them, 19,957 queries in the test set do not appear in the training set, and they are considered as zero-shot case samples.

3.2 Experimental Setup

The pretrained T5 (t5-base) is our backbone model [36]. We use AdamW as an optimizer with a learning rate of {2e-5, 5e-5} for the training, and adopt batch size = 128 in all settings. In our experiments with prefix adaptor, we use a prefix length of 5. The hidden states dimension of the MLP layer is 512, and the dropout is 0.1 in all settings. To construct the hierarchical document ids, k-means ($k = 10$) clustering provided by `fast-kmeans` [1] is applied recursively over all document embeddings generated from the pre-trained BERT (bert-base-uncased) model. All documents are clustered into 10 clusters. The algorithm is applied recursively if the cluster size exceeds 100. Clusters with sizes smaller than 100 documents are assigned an arbitrary unique number from 0-99 as the cluster id. The next level's cluster id is appended to the current level. The average document id length is 6. The threshold in Algorithm 1 and s are 0.6, and 0.8, respectively. For the weights in Function (2), w_1 is 3/6, w_2 is 2/6, w_3 is 1/6.

3.3 Experimental Results and Analysis

Experiment results between PersonalTM and several baseline methods are presented in Table 1. We use p@k and Mean Reciprocal Rank (MRR) as our evaluation metrics, where p@k measures the accuracy of the top k documents for the incoming query, MRR

[1] https://pypi.org/project/fast-pytorch-kmeans/

calculates the mean of the inverse of the ranks at which the first relevant document is retrieved for the incoming query. Both of them are in %. Consistent with the observations in [32], DSI significantly outperforms BM25 and finetuned DE (encoder in Finetuned DE is the same as the encoder of finetuned DSI). With our proposed hierarchical loss function, p@1 increases by 0.66% compared to the one with cross-entropy loss only (3 - 4 in Table 1). It is because hierarchical loss grants higher penalties for the error of higher-level digits, it improves the precision. To further investigate how the hierarchical loss contributes to the performance, we find that hierarchical loss improves the precision at every document id digit, especially at the first three digits, with an average p@1 increment of 4.83%.

#	Method	p@1	p@10	MRR
1	BM25	21.61	32.87	25.46
2	Finetuned DE	30.58	42.13	34.25
3	DSI	67.42	75.84	70.58
4	DSI + HieLoss	68.08	77.33	71.52
5	DSI + UserIdentifier	69.16	77.60	72.19
6	DSI + HieLoss + UserIdentifier	70.06	78.47	73.04
7	PersonalTM (h in latest session)	71.20	80.10	74.25
8	PersonalTM (h relevant to q)	79.60	87.47	82.51

Table 1: Results of baseline methods and PersonalTM. Note all baselines and treatments are finetuned with the same training dataset.

By comparing lines 3 (no user identifier p) and 5 (with user identifier p) in Table 1, p@1 is improved by 1.74%. This indicates p can effectively help the model remember user's preference, particularly for predictions of unseen queries from the existing users.

Comparison of lines 6 and 7 in Table 1 demonstrates the strength of the proposed model structure involving personal context h. By only using h from the latest session, p@1 can be improved by 1.14%. By comparing line 8 with lines 6 and 7, we can conclude that after applying the similarity selection to denoise personalized context, p@1 is significantly improved by 9.54% and 8.40% respectively. We also compare the performance with other personalization methods proposed in previous work in Table 2. PersonalTM has a comparable or better performance compared to other personalization methods.

Method	p@1	Method	p@1	Method	p@1
P-Click [1]	59.56	GRADP [40]	77.06	SLTB [12]	71.09
HRNN [13]	76.53	PersonalTM	79.60		

Table 2: Comparison with other personalization methods. precision numbers are averaged reported values in [15].

The results of the prefix adaptor are shown in Table 3. By comparing lines 1 - 2, the model parameters and training time of the adaptor are reduced by 10x and 2x compared to fine-tuning, respectively. It significantly reduces cost and improves agility for index updates while achieving comparable performance.

The performance of relevant personal context similarity measurements is shown in lines 3.1 and 4.1 in Table 3. Comparing 3.1 and 3.2 as well as 4.1 and 4.2, we find that our proposed PersonalTM beats the naive method that simply uses the latest clicked document by 1.09% and 1.62% for p@1, respectively. It is because our model structure explicitly learns the mapping between the query and the personal context. Besides, the proposed PersonalTM forces the decoder to digest this information without diluting it by the long forwarding network. The result of fusion performance is shown in line 4.3 of Table 3. It improves p@1 by 2.16% compared to PersonalTM without fusion. Note fusion increases the computation time by an additional 30%, especially during the decoding process. Therefore, there is trade-offs between the computation cost, model size or storage space, and performance.

#	Method	Imple	p@1	p@10	MRR	para
1	w/o h	finetune	70.06	78.47	73.04	222.9
2	w/o h	prefix	69.90	78.25	72.87	29.5
Using latest clicked document						
3.1	$[p; q; h]$	prefix	68.92	77.83	72.10	29.5
3.2	PersonalTM	prefix	70.01	78.53	73.01	29.5
Relevant personal context similarity measurement						
4.1	$[p; q; h]$	prefix	75.26	85.37	79.28	29.5
4.2	PersonalTM	prefix	76.89	86.06	80.20	29.5
4.3	PersonalTM (f)	prefix	79.05	86.98	82.10	29.7

Table 3: Results with prefix adaptor.

#	Method	p@1	p@10	MRR
a	BM25 (with relevant history)	20.97	31.03	24.53
b	DualEncoder (T5)	6.66	17.81	9.90
c	DSI	8.14	16.76	10.71
d	DSI + HieLoss	11.45	23.39	15.14
e	DSI + HieLoss + UserIdentifier	15.17	28.20	19.21
f	PersonalTM (prefix adaptor)	37.19	56.94	43.76

Table 4: Evaluation on Zero-shot cases.

Furthermore, we evaluate the model performance on the zero-shot set, whose queries are never seen in the past. The results are shown in Table 4. We found that DSI's zero-shot performance is not ideal, which aligns with the observation in [32]. But by comparing line f with the rest, we achieve significant improvement by using PersonalTM and the relevant personal context.

4 CONCLUSION

In this work, we propose a transformer memory framework for personalized retrieval, integrating two essential types of personalized information into the model. We propose a novel decoder architecture that effectively leverages personalized context without increasing trainable parameters, and apply a hierarchical loss function to optimize the performance. Our framework also incorporates the prefix adaptor mechanism to facilitate the learning of the interaction between the query and contextual personalized features. In the future, we will further enhance the design by incorporating incremental learning and enabling dynamic updates to overcome the limitations of frequent index updates and zero-shot retrieval.

REFERENCES

[1] Paul N Bennett, Ryen W White, Wei Chu, Susan T Dumais, Peter Bailey, Fedor Borisyuk, and Xiaoyuan Cui. 2012. Modeling the impact of short-and long-term behavior on search personalization. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. 185–194.

[2] Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Wen-tau Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. *arXiv preprint arXiv:2204.10628* (2022).

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[4] Chun-Fu Richard Chen, Quanfu Fan, and Rameswar Panda. 2021. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *Proceedings of the IEEE/CVF international conference on computer vision*. 357–366.

[5] Yinpeng Chen, Xiyang Dai, Dongdong Chen, Mengchen Liu, Xiaoyi Dong, Lu Yuan, and Zicheng Liu. 2022. Mobile-former: Bridging mobilenet and transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5270–5279.

[6] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. 2022. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534* (2022).

[7] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. 2022. Mixformer: End-to-end tracking with iterative mixed attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13608–13618.

[8] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2020. Autoregressive entity retrieval. *arXiv preprint arXiv:2010.00904* (2020).

[9] Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W Bruce Croft. 2017. Neural ranking models with weak supervision. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*. 65–74.

[10] Yang Deng, Yaliang Li, Wenxuan Zhang, Bolin Ding, and Wai Lam. 2022. Toward Personalized Answer Generation in E-Commerce via Multi-perspective Preference Modeling. *ACM Transactions on Information Systems (TOIS)* 40, 4 (2022), 1–28.

[11] 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).

[12] Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. 2007. A large-scale evaluation and analysis of personalized search strategies. In *Proceedings of the 16th international conference on World Wide Web*. 581–590.

[13] Songwei Ge, Zhicheng Dou, Zhengbao Jiang, Jian-Yun Nie, and Ji-Rong Wen. 2018. Personalizing search results using hierarchical RNN with query-aware attention. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 347–356.

[14] Daniel Gillick, Alessandro Presta, and Gaurav Singh Tomar. 2018. End-to-end retrieval in continuous space. *arXiv preprint arXiv:1811.08008* (2018).

[15] Qian Guo, Wei Chen, and Huaiyu Wan. 2021. AOL4PS: A large-scale data set for personalized search. *Data Intelligence* 3, 4 (2021), 548–567.

[16] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366* (2021).

[17] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*. PMLR, 2790–2799.

[18] Kai Hui, Honglei Zhuang, Tao Chen, Zhen Qin, Jing Lu, Dara Bahri, Ji Ma, Jai Prakash Gupta, Cicero Nogueira dos Santos, Yi Tay, et al. 2022. Ed2lm: Encoder-decoder to language model for faster document re-ranking inference. *arXiv preprint arXiv:2204.11458* (2022).

[19] Taegwan Kang, Hwanhee Lee, Byeongjin Choe, and Kyomin Jung. 2021. Entangled bidirectional encoder to autoregressive decoder for sequential recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1657–1661.

[20] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906* (2020).

[21] Hyunji Lee, Sohee Yang, Hanseok Oh, and Minjoon Seo. 2022. Generative Retrieval for Long Sequences. *arXiv preprint arXiv:2204.13596* (2022).

[22] Jiwei Li, Michel Galley, Chris Brockett, Georgios P Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. *arXiv preprint arXiv:1603.06155* (2016).

[23] Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190* (2021).

[24] Ruixue Lian, Che-Wei Huang, Yuqing Tang, Qilong Gu, Chengyuan Ma, and Chenlei Guo. 2022. Incremental user embedding modeling for personalized text classification. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 7832–7836.

[25] Jiongnan Liu, Zhicheng Dou, Qiannan Zhu, and Ji-Rong Wen. 2022. A Category-aware Multi-interest Model for Personalized Product Search. In *Proceedings of the ACM Web Conference 2022*. 360–368.

[26] Jingjing Liu, Chang Liu, and Nicholas J Belkin. 2020. Personalization in text information retrieval: A survey. *Journal of the Association for Information Science and Technology* 71, 3 (2020), 349–369.

[27] Fatemehsadat Mireshghallah, Vaishnavi Shrivastava, Milad Shokouhi, Taylor Berg-Kirkpatrick, Robert Sim, and Dimitrios Dimitriadis. 2021. UserIdentifier: implicit user representations for simple and effective personalized sentiment analysis. *arXiv preprint arXiv:2110.00135* (2021).

[28] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. WebGPT: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332* (2021).

[29] Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B Hall, Daniel Cer, and Yinfei Yang. 2021. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. *arXiv preprint arXiv:2108.08877* (2021).

[30] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* 21, 140 (2020), 1–67.

[31] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems* 27 (2014).

[32] Yi Tay, Vinh Q Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. 2022. Transformer memory as a differentiable search index. *arXiv preprint arXiv:2202.06991* (2022).

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[34] Elena Voita, Rico Sennrich, and Ivan Titov. 2019. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. *arXiv preprint arXiv:1909.01380* (2019).

[35] Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Hao Sun, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, et al. 2022. A Neural Corpus Indexer for Document Retrieval. *arXiv preprint arXiv:2206.02743* (2022).

[36] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771* (2019).

[37] Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I Wang, et al. 2022. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *arXiv preprint arXiv:2201.05966* (2022).

[38] Han Zhang, Songlin Wang, Kang Zhang, Zhiling Tang, Yunjiang Jiang, Yun Xiao, Weipeng Yan, and Wen-Yun Yang. 2020. Towards personalized and semantic retrieval: An end-to-end solution for e-commerce search via embedding learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2407–2416.

[39] Giulio Zhou and Jacob Devlin. 2021. Multi-Vector Attention Models for Deep Re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 5452–5456.

[40] Y Zhou, Z Dou, S Ge, and JR Wen. 2019. Dynamic personalized search based on RNN with attention mechanism. *Chinese Journal of Computer* 42 (2019), 812–826.

[41] Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. 2020. Encoding history with context-aware representation learning for personalized search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1111–1120.

[42] Shengyao Zhuang, Houxing Ren, Linjun Shou, Jian Pei, Ming Gong, Guido Zuccon, and Daxin Jiang. 2022. Bridging the gap between indexing and retrieval for differentiable search index with query generation. *arXiv preprint arXiv:2206.10128* (2022).

[43] Simiao Zuo, Qingyu Yin, Haoming Jiang, Shaohui Xi, Bing Yin, Chao Zhang, and Tuo Zhao. 2022. Context-Aware Query Rewriting for Improving Users' Search Experience on E-commerce Websites. *arXiv preprint arXiv:2209.07584* (2022).