Enhancing User Behavior Sequence Modeling by Generative Tasks for Session Search

Haonan Chen  
Zhicheng Dou  
Gaoling School of Artificial Intelligence, Renmin University of China  
Beijing, China  
hncchen@ruc.edu.cn  
dou@ruc.edu.cn

Yutao Zhu  
University of Montreal  
Montreal, Quebec, Canada  
yutaozhu94@gmail.com

Zhao Cao  
Xiaohua Cheng  
Poisson Lab, Huawei  
Beijing, China  
caozhao1@huawei.com  
chengxiaohua1@huawei.com

Ji-Rong Wen  
Beijing Key Laboratory of Big Data Management and Analysis Methods, Beijing China  
jrwen@ruc.edu.cn

ABSTRACT

Users’ search tasks have become increasingly complicated, requiring multiple queries and interactions with the results. Recent studies have demonstrated that modeling the historical user behaviors in a session can help understand the current search intent. Existing context-aware ranking models primarily encode the current session sequence (from the first behavior to the current query) and compute the ranking score using the high-level representations. However, there is usually some noise in the current session sequence (useless behaviors for inferring the search intent) that may affect the quality of the encoded representations. To help the encoding of the current user behavior sequence, we propose to use a decoder and the information of future sequences and a supplemental query. Specifically, we design three generative tasks that can help the encoder to infer the actual search intent: (1) predicting future queries, (2) predicting future clicked documents, and (3) predicting a supplemental query. We jointly learn the ranking task with these generative tasks using an encoder-decoder structured approach. Extensive experiments on two public search logs demonstrate that our model outperforms all existing baselines, and the designed generative tasks can actually help the ranking task. Besides, additional experiments also show that our approach can be easily applied to various Transformer-based encoder-decoder models and improve their performance.

CCS CONCEPTS
• Information systems → Retrieval models and ranking.

KEYWORDS

Auto-session-encoder, Session Search, Document Ranking

ACM Reference Format:
Haonan Chen, Zhicheng Dou, Yutao Zhu, Zhao Cao, Xiaohua Cheng, and Ji-Rong Wen. 2022. Enhancing User Behavior Sequence Modeling by Generative Tasks for Session Search. In Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM ’22), October 17–21, 2022, Atlanta, GA, USA. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3511808.3557310

1 INTRODUCTION

With the development of search engines, users’ information needs have become increasingly complex. A user usually issues several queries and examines some documents to complete a search task. These user behaviors (e.g., issued queries and clicked documents) that occur during a relatively brief period are referred to as a search session [2, 14, 32, 36]. Modeling the session context has been demonstrated to be beneficial for understanding search intent [2, 36].

Several early studies have attempted to model session context based on statistical techniques, which inevitably neglect some valuable features [3, 12, 30, 33]. With the emergence of deep learning, many neural context-aware ranking models have been proposed [1, 2, 6, 25, 35]. They use recurrent neural networks (RNNs) to encode user behaviors into latent representation [1, 2, 6], or pre-trained language models (PLMs), such as BERT [10], to get able features [3, 12, 30, 33]. This representation is used to compute ranking scores. However, the current session sequence (from the beginning to the current query) may contain some useless information (i.e., noise) that could cause the encoders (e.g., BERT or RNNs) to misinterpret the real user intent. Figure 1 illustrates an example where a user has issued several queries and examined some documents. The current query is "Hulu", and the user is trying to download the iOS version of the Hulu application. Evidently, the previous query "NBA live" is useless for inferring the current search intent, and simply encoding
such noise may degrade the sequence’s representation. Unfortunately, the quality of the session representation, i.e., whether the real user intent has been encoded, has received little attention in existing studies.

A straightforward way to tackle this issue is to apply the auto-encoder technique. Auto-encoder has an encoder-decoder structure, where the decoder is used to recover the input sequence based on the representation computed by the encoder. The encoder is thereby trained to capture the most important information from the input sequence. However, applying auto-encoder to session search is non-trivial in three aspects: (1) Generating the whole session sequence is challenging. Typically, a session consists of many queries and documents, which are too long for the decoder to recover. Besides, most auto-regressive decoders, including those pre-trained on large-scale corpora (e.g., GPT [26]), are incapable of modeling the relevance between queries and documents [20]. (2) There is noise in the session sequence. As indicated above, a session sequence usually contains user behaviors irrelevant to the current information need. Thus, the decoder should not generate all behaviors in the session sequence without differentiation. (3) The user behaviors that represent the real search intent may be implicit in the current session sequence. A user’s current information need is often complicated and cannot be described clearly by the user (or understood by the search engine). For example, the behavior reflecting the user’s information need may be a future query in the session or a similar query issued by a different user in another session (i.e., another user can successfully address the same information need while this user cannot). Under this circumstance, simply recovering the current session sequence is ineffective.

To address these problems, we employ an encoder-decoder structure and design several generative tasks specifically for session search to assist the encoder in inferring the search intent more accurately. Specifically, we design three generative tasks:

**Task 1**: Predicting future queries. As the session progresses, the user becomes more explicit about their actual information need. Thus, subsequent queries within the same session can more accurately reflect the search intent.

**Task 2**: Predicting future clicked documents. In addition to future queries, we also consider future user clicks because the documents usually contain more detailed information than keyword-based queries.

**Task 3**: Predicting a supplemental query. As explained in the third problem above, some queries in other users’ sessions may be helpful in understanding the current search intent.

All of these generative targets are more accurate (or supplemental) descriptions of the current search intent. Therefore, only if the encoder has successfully encoded the user’s search intent into the representation can the decoder predict these sequences using the representation of the current user behavior sequence. Besides, our designed generative tasks can address the three aforementioned challenges as follows: For the first problem, Task 1 & 2 attempt to generate the future queries and documents separately, so avoiding generating long sequences or modeling relevance between queries and documents, making the generation easier. For the second problem, we explore many potential generation targets and propose these three tasks that can actually help the encoder infer actual search intent. Our experiments in Section 5.3 will show the effectiveness of these generative tasks. For the third problem, all these tasks try to predict future sequences (or a supplemental query), i.e., information that is not in the current sequence.

We propose to jointly learn the ranking and generative tasks by an encoder-decoder structured approach. Specifically, we attempt to use future sequences and a supplemental query as generation targets to enhance the encoder’s ability to represent session context. We call our model ASE – Auto-Session-Encoder, which is based on a pre-trained BART [16]. Experimental results on two public search logs (AOL [24] and Tiangong-ST [7]) show that ASE outperforms existing methods, which demonstrates its effectiveness. Moreover, the consistent performance improvements on top of different Transformer-based encoder-decoder models demonstrate our approach’s effectiveness and wide applicability.

To summarize, the contributions of this work are as follows: (1) We propose Auto-Session-Encoder, which employs several generative tasks to explicitly enhance the ability to encode a user behavior sequence under an encoder-decoder framework. (2) We design three generative tasks to utilize the future sequences and a supplemental query to train a better representation of the current session sequence. Experimental results demonstrate the effectiveness of the generative tasks. (3) We demonstrate that our model can be easily adapted to various Transformer-based encoder-decoder models other than BART, indicating its wide applicability.

## 2 RELATED WORK

### 2.1 Session Search

There are already some traditional approaches that utilize session context to infer search intent [3, 4, 12, 30, 33]. Specifically, Shen et al. [30] used statistical language models to combine session context and the current query for better ranking performance. Van Gysel et al. [12] explored lexical query modeling for session search. They found that specialized session search methods are more suitable for modeling long sessions than naive term weighting methods.

With the emergence of deep learning, researchers have focused on designing neural context-aware ranking models [1, 2, 6, 25, 36, 37]. Specifically, Ahmad et al. [1, 2] encoded queries and documents using RNNs and attention mechanism. Then they jointly learned the ranking task and query suggestion task. Xu et al. [25] concatenated the current session sequence and put them into a BERT encoder. Then they applied a hierarchical behavior-aware attention module over the BERT encoder to get high-level representations for ranking. Zuo et al. [37] modeled multi-granularity historical query change. They obtained multi-level representations of the session using Transformer-based encoders. Chen et al. [6] integrated representation and interaction. They encoded the session history into a latent representation and used it to enhance the current query and the candidate document. Then they captured the interaction-based information between the enhanced query and the candidate document. Zhu et al. [36] utilized data augmentation and contrastive learning to pre-train a BERT encoder that can represent the session sequence better. Most existing models use an encoder to model the current session sequence and obtain high-level representations of the sequence to compute ranking scores. However, because of
noise in the session, the representations may fail to encode the user’s actual search intent. Our model aims to enhance user behavior sequence modeling by multiple designed generative tasks that attempt to utilize future sequences and a supplemental query. By this, we attempt to explicitly ensure the actual search intent has been encoded into the high-level representations.

2.2 Generative Tasks for IR

There are already some works trying to utilize generative tasks to improve retrieval performance [1, 2, 9, 18, 21]. Liu et al. [18] demonstrated that generative tasks can make retrieval modeling more generalized. Mao et al. [21] showed that generating heuristically discovered relevant contexts for queries can improve their retrieval and QA results. Cheng et al. [9] utilized the next query prediction task to help personalized re-ranking. Ahmad et al. [1, 2] illustrated that the query suggestion task could improve the ranking quality of session search. Though they have already demonstrated the effectiveness of predicting the next query, there are more generation targets to be explored. Specifically, we find that predicting future queries (not only the next query), predicting future clicked documents, and predicting a supplemental query can all help model user behavior sequences. After exploring various potential generative targets, we design multiple generative tasks specifically for session search (Section 3.4) to help model the current session context.

3 AUTO-SESSION-ENCODER

Session search aims to utilize the user behavior sequence to rank candidate documents. Most existing models use an encoder to model session context and get high-level representations of the sequence to compute ranking scores. However, the representations may lack the information of the user’s actual search intent because of noisy user behaviors. Our model aims to enhance the ability of the encoder with three designed generative tasks that attempt to utilize the information of future sequences and a supplemental query. By this, we try to help the encoder to encode the actual search intent into the high-dimensional representations of the session sequence.

3.1 Problem Definition

Before shedding light on our proposed model, we will state some notations about session search. Suppose a query \( q_i \) has \( M \) clicked documents \( D_i = \{d_{i,1}, d_{i,2}, \ldots, d_{i,M}\} \). Following [25, 36, 37], we keep the first clicked document for each historical query to construct the sequence. Then the current session sequence \( S \) when the user is issuing the \( n \)-th query \( q_n \) can be denoted as:

\[
S = \{ (q_1, d_1), (q_2, d_2), \ldots, (q_n) \}.
\]

The goal of session search (or context-aware document ranking) is to model the contextual information to obtain the ranking scores of the candidate documents \( D_c \) and rank them accordingly. We will focus on how to get the score of a candidate document \( d_c \) in the rest of the paper. Note that the current session sequence \( S \) only contains the historical and present user behaviors when a user is issuing \( q_n \). However, we will utilize future sequences and a supplemental query as generation targets while training.

3.2 Overall Structure

In this part, we will introduce the overall structure of ASE. ASE jointly learns the ranking task and the generative tasks as follows:

1. **Ranking.** As shown in the left part of Figure 2, we attempt to compute the ranking score of the candidate document \( d_c \) for the ranking task. To model the session context, ASE first concatenates the session sequence \( S \) with \( d_c \) and puts it into the encoder. Then ASE gets the output of the "[CLS]" token as the high-dimensional representation. Finally, we apply a linear projection on this representation to get the ranking score of \( d_c \) (Section 3.3).

2. **Generation.** As shown in Figure 3 and the right part of Figure 2, we aim to enhance the ability of the encoder using the decoder and three generative tasks (Section 3.4). These generative tasks are comprised of (i) predicting future queries, (ii) predicting future clicked documents, and (iii) predicting a supplemental query. Finally, by jointly learning the ranking task and the generative tasks (Section 3.5), the encoder can model user behavior sequences better and learn representations that contain actual search intent. Note that these generative task are only used in the training stage, for enhancing the representation ability of the encoder. At inference time, we will only use the enhanced encoder to score the candidate documents.

In this work, we choose the pre-trained language model BART [16] as ASE’s backbone because: (1) BART is a Transformer-based [31] encoder-decoder model with a bidirectional (BERT-like) encoder and an autoregressive (GPT-like) decoder. We can naturally implement our ranking and designed generative tasks on this model. (2) BART utilizes self-supervised pre-training, which makes it perform very well on many generative tasks and do not reduce performance on discriminative tasks [16]. (3) BART-base model uses six layers in the encoder and decoder, respectively, which makes it contain the comparable number of parameters as BERT-base model (twelve layers in the encoder). Besides, the number of training steps and the data used for pre-training BART is the same as BERT. Thus, we choose BART as ASE’s backbone for fair comparisons with BERT-based baseline models [25, 36]. In addition, as demonstrated in Section 5.5, we can easily apply ASE to other Transformer-based encoder-decoder models.

3.3 Modeling the Current Session Sequence

In this section, we will illustrate how ASE uses the encoder to model session context. As shown in the left part of Figure 2, ASE first puts the concatenated sequence into the encoder and gets a high-level representation from the "[CLS]" token. Then ASE makes it go through a linear projection to get the ranking score.

Following previous works [25, 36], we treat the ranking task as a sequence pair classification problem. Given the session sequence \( S \), we consider the current user behaviors as one sequence \( \{q_1, d_1, q_2, d_2, \ldots, q_n\} \), and the candidate document to be scored as another sequence \( \{d_c\} \). After adding some special tokens, the constructed input sequence is as follows:

\[
I = [\text{CLS}]q_1[\text{EOS}]d_1[\text{EOS}] \cdots q_n[\text{EOS}][\text{SEP}]d_c[\text{EOS}][\text{SEP}],
\]

where "[CLS]" is the classification token, "[EOS]" is used to identify the end of each query and document, "[SEP]" is used to separate the sequence pair for classification.
Figure 2: The diagram of ASE. Training: For ranking, ASE concatenates the current user behavior sequence and puts it into the encoder. Then ASE makes the output of the "[CLS]" token go through an MLP to get the ranking score of $d_c$. For each generative task, ASE takes the generative target as the generation label of the decoder. With the ranking loss and generation losses ready, ASE jointly learns these tasks to enhance the encoder. Inference: ASE only uses the encoder to rank the candidate documents.

Figure 3: The three generative tasks designed for session search. They are only used when training. We take a session that has five query-document pairs as an example. Suppose $q_2$ is the current query, then our goal is to utilize the information of future sequences and a supplemental query to model the current user behavior sequence. The queries and documents that we believe can help infer the user’s current search intent are marked with the color red.

Then ASE makes $I$ go through the encoder and takes the output of "[CLS]" token as the high-level representation:

$$R = \text{Encoder}(I)_{[CLS]}.$$  

Finally, we use a multi-layer perceptron (MLP) on this representation to get the ranking score:

$$\text{Score}(d_c) = \text{MLP}(R).$$  

3.4 Enhancing Encoder with Generative Tasks

As stated in Section 1, the current session sequence $S$ may contain some noise that is irrelevant to the current search intent, which may affect the quality of the session representation $R$. Thus, we propose to enhance the encoder’s ability with a decoder and three generative tasks that are designed for session search. Specifically, we attempt to utilize the information of future sequences and a supplemental query. This can help the encoder to encode the actual search intent into $R$.

As presented in Figure 3, we take a session that has five query-document pairs as an example to illustrate the three designed generative tasks. Suppose during training, $q_3$ “Hulu” is the current query, and there are some candidate documents to be ranked. Then the current user behavior sequence is $\{q_1, d_1, q_2, d_2, q_3\}$. Let us suppose that the user is trying to download the application Hulu from App Store. However, the current sequence has some noise (queries and documents that are not marked red). For example, $q_1$ “NBA Live” may mislead the search system to infer that the user is trying to watch live NBA games on the Hulu website.

Since the current session is noisy, the encoder may have trouble encoding the actual search intent into $R$. We propose to utilize generative tasks to help the encoding of the current sequence $S$ during training. Specifically, we design three generative tasks for session search as follows:

1. Predicting Future Queries. The user often tries to issue a new query when the current one does not satisfy her information need. Besides, as the searching progresses, the user has a more explicit understanding of the search task, and the quality of the queries they issue is also getting higher. Thus, the subsequent queries in the session can often represent the search intent better than the current query. For example, if ASE can predict $q_4$ “How to use Hulu App” given the information of $S$, then we believe the encoder has encoded the user’s search intent (download the Application Hulu from App Store) into the high-level representations. To utilize the information of the future queries, we concatenate them as a
We will notice that the generation starts at the current clicked
when the session sequence is noisy. Besides, if the user fails to
want to confuse our model with too many different topics.

(2) Predicting Future Clicked Documents. Similar to the
reasons above, the future clicked documents (especially the current
query’s click, which we also consider a future clicked document)
often contain valuable information about the user’s search interest.
Besides, we can usually get more detailed information from doc-
ments than queries since many queries contain only keywords.
For example, d₃ “Hulu iOS Download” is obviously a more specific
and accurate version of q₃. It would be great if the decoder could
predict this information from the high-dimensional representations
of S. Thus, we believe the future clicked documents can also help
the encoder infer the search intent, and we utilize their information
as follows:

\[ GT_2 = d_n[SEP]d_{n+1}[SEP] \ldots d_{n+w-1}[SEP], \]

We will notice that the generation starts at the current clicked
document dₙ since it is considered a future sequence.

(3) Predicting a Supplemental Query. Many queries contain
only keywords, making them hard to be understood, especially
when the session sequence is noisy. Besides, if the user fails to
address her search task, then the user behavior that can represent
her search intent may be implicit in the current session. For example,
the query from another session q₃’ “Hulu App Store” can supplement
our model’s understanding of S. Following previous works [6, 19],
we attempt to find a query to supplement the information of the
current query, which can make our model more robust. We treat
the training data as the query database, and for each query, we
mine one query that we believe contains supplemental information
from the database. We use the equation suggested by Chen et al. [6]
to measure the supplemental rate of the query we choose:

\[ \text{sup}(q'_n, q_n) = \text{spe}(q'_n, q_n) + \text{sim}(q'_n, q_n), \]

where q'_n is the candidate query in the database, \( \text{sup}(q'_n, q_n) \) is its
supplemental rate; \( \text{spe}(q'_n, q_n) = \frac{\text{len}(q'_n) - \text{len}(q_n)}{\text{len}(q_n)} \)
every word of q_n appear in q'_n, otherwise it is 0. This component computes the
specificity between q'_n and q_n; \( \text{sim}(q'_n, q_n) \) is the similarity
between q'_n and q_n, which is computed by the python class Se-
quenceMatcher.¹

We choose the query q'_n that has the highest supplemental rate
to be our last generation target:

\[ GT₃ = q'_n[SEP]. \]

Different from GT₁ and GT₂, we only use one sequence here as the
generation target. This is because the queries mined from other
sessions often represent different information needs, and we do not
want to confuse our model with too many different topics.
For those queries and documents in GT₁ and GT₂ that are empty
(lacking future information or recording error in the datasets), we use
"[empty_q]\" and "[empty_d]\" to pad them respectively.

With these generation targets ready, we treat them as different
generative tasks and train the decoder to generate them separately
during training. If the decoder could predict these targets from the

¹https://docs.python.org/3/library/difflib.html
4 EXPERIMENTAL SETUP

4.1 Datasets and Evaluation Metrics

4.1.1 Datasets. We conduct our experiments on AOL search log [24] and Tiangong-ST search log [7]. They are both public large-scale search logs. We have also considered MS MARCO Conversational Search dataset. However, the sessions of this dataset are artificial, and we want to study actual user behaviors from real-world search logs. Therefore, we do not use this dataset and stick to AOL and Tinagong-ST which are widely used in existing works.

We process the AOL search log following Ahmad et al. [2]. Each query of the training and validation sets contains five candidate documents, and each one of the test set contains 50 candidates that are retrieved by the BM25 algorithm [28].

Tiangong-ST [7] is collected from a Chinese commercial search engine. For the last query of each in the test set, its candidate documents have human-annotated relevance labels (0 to 4). As suggested by the original paper of this dataset [7], we will use the queries that have relevance labels when testing. For more details on this dataset, please refer to [7].

Following previous works [2, 6, 25, 36, 37], we only use the title of each document as its content. The statistics of these two datasets are presented in Table 1.

4.1.2 Evaluation Metrics. Following previous works [2, 6, 25, 36, 37], we use Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), and Normalized Discounted Cumulative Gain (NDCG) at position $k$ (NDCG@$k$, $k = 1, 3, 5, 10$) as metrics. We use TREC’s evaluation tool (trec_eval) [11] to compute all evaluation results.

4.2 Baselines

Following previous works [6, 36, 37], we compare ASE with two kinds of baselines:

(1) Ad-hoc ranking models only use the information of $q_n$ and $d_c$ to get the ranking score.
   • BM25 [28] is a traditional ranking algorithm based on the probabilistic retrieval framework. It treats the relevance between two behaviors interact with each other, it first uses the representation of session history to enhance $q_n$ and $d_c$. Then it makes the enhanced $q_n$ and $d_c$ interact on the word level. • BERT [10] and BART [16] are the vanilla versions of BERT-base and BART-base.
   We include these two as baselines to demonstrate that it is fair (Section 3.2) to compare our BART-based model ASE to BERT-based baselines (HBA, COCA). When fine-tuning these two models, we simply concatenate $S$ with $d_c$ and put it into the encoder (we do not use the decoder of BART). Then we make the output of [CLS] go through an MLP to get the ranking score. • COCA [36] utilizes data augmentation and contrastive learning to pre-train a BERT encoder that can represent the session sequence better. It is the state-of-the-art model which has been demonstrated effective in NTCIR-16 Session Search Track [5, 8].

4.3 Implementation Details

For AOL, we use the BART-base model provided by the authors of [16] on Huggingface. For Tiangong-ST, we use the Chinese BART-base model provided by the authors of [29] on Huggingface. Following T5 [27], we use a unique task identifier at the beginning of the input sequence for each task. Following previous works [25, 36], we truncate the sequence from the head if its length exceeds the maximum length.

For details of the instantiations, one can refer to our code.5

5 RESULTS AND ANALYSIS

5.1 Overall Results

The overall performances of all models are presented in Table 2. The results show that context-aware ranking models generally perform better than ad-hoc ones, which indicates the effectiveness of modeling session context. Besides, we can further obtain the following observations:

(1) ASE outperforms all baselines in terms of all metrics on both datasets. Specifically, ASE outperforms COCA, a strong baseline that utilizes pre-training and data augmentation strategies. It demonstrates the effectiveness of utilizing our generative tasks to model the current search intent. In future work, we will try to incorporate pre-training techniques (e.g., contrastive learning) and data augmentation strategies (e.g., curriculum learning) into ASE to further improve its performance. Besides, we will notice that the improvements of ASE on AOL are more significant than those on Tiangong-ST. The potential reasons are as follows: (i) The base performances on Tiangong-ST are already very high because there are more than 77.4% candidate documents with relevance scores that are larger than 1 in the test set. Specifically, even the BM25 algorithm can achieve 0.8541 in terms of NDCG@10 on this dataset. Therefore, it is more difficult for ASE to improve its performance on Tiangong-ST. This phenomenon has also been noticed by Zhu et al. [36]. (ii) As suggested by the authors of this dataset [7], we use the queries that have human-annotated relevance labels when testing, and all of them are the last ones in their sessions.

5https://github.com/haon-chen/ASE-Official

4https://huggingface.co/facebook/bart-base
5https://github.com/haon-chen/ASE-Official
6https://github.com/microsoft/MSMARCO-Conversational-Search
However, as we find in Section 5.6, ASE can give more considerable improvements on the queries with fewer histories. This is because ASE utilizes future sequences and a supplemental query to train the encoder to predict the actual search intent.

(2) The vanilla version of the BART model underperforms that of BERT. For example, BERT and BART achieve about 0.3990 and 0.3908 in terms of NDCG@1 on AOL dataset, respectively. This demonstrates that the original encoder of BART performs worse than BERT’s encoder, which makes the comparisons of ASE and BERT-based baselines (HBA, COCA) fair (as illustrated in Section 3.2). We believe the reason is that BERT-base has 12 layers, whereas the encoder of BART-base only has 6 layers. However, ASE can still outperform the BERT-based baselines based on a worse backbone than BERT (for the encoder), which further demonstrates its effectiveness.

### Table 2: Overall results on AOL and Tiangong-ST.

| Dataset | Model | NDCG@1 | @3 | @5 | @10 | MAP | MRR |
|---------|-------|--------|----|----|-----|-----|-----|
| **AOL** | **Model** | **NDCG@1** | **@3** | **@5** | **@10** | **MAP** | **MRR** |
| **Ad-hoc Ranking Models** | | | | | | | |
| BM25 | 0.1189 | 0.1862 | 0.2136 | 0.2481 | 0.2220 | 0.2271 |
| ARC-I | 0.1983 | 0.3108 | 0.3499 | 0.3951 | 0.3361 | 0.3475 |
| ARC-II | 0.2428 | 0.3564 | 0.4026 | 0.4486 | 0.3834 | 0.3951 |
| KNRM | 0.2397 | 0.3868 | 0.4322 | 0.4761 | 0.4038 | 0.4133 |
| Duet | 0.2492 | 0.3822 | 0.4246 | 0.4675 | 0.4008 | 0.4111 |
| **Context-aware Ranking Models** | | | | | | | |
| CARS | 0.2836 | 0.4117 | 0.4542 | 0.4971 | 0.4297 | 0.4408 |
| HBA | 0.3773 | 0.5241 | 0.5624 | 0.5971 | 0.5281 | 0.5384 |
| RICR | 0.3894 | 0.5267 | 0.5648 | 0.5971 | 0.5338 | 0.5450 |
| HCQN | 0.3990 | 0.5441 | 0.5783 | 0.6070 | 0.5488 | 0.5549 |
| BART | 0.3908 | 0.5414 | 0.5797 | 0.6108 | 0.5450 | 0.5511 |
| BERT | 0.3990 | 0.5440 | 0.5818 | 0.6123 | 0.5471 | 0.5572 |
| COCA | 0.4024 | 0.5478 | 0.5849 | 0.6106 | 0.5500 | 0.5601 |
| ASE | **0.4144** | **0.5682** | **0.6007** | **0.6283** | **0.5650** | **0.5752** |

| **Tiangong-ST** | **Model** | **NDCG@1** | **@3** | **@5** | **@10** | **MAP** | **MRR** |
| **Model** | **NDCG@1** | **@3** | **@5** | **@10** | **MAP** | **MRR** |
| **Ad-hoc Ranking Models** | | | | | | | |
| BM25 | 0.6160 | 0.6123 | 0.5808 | 0.5752 | 0.5572 | 0.5551 |
| ARC-I | 0.7612 | 0.7581 | 0.7639 | 0.7695 | 0.7552 | 0.7532 |
| ARC-II | 0.7670 | 0.7636 | 0.7704 | 0.8094 | 0.7873 | 0.7893 |
| HCQN | 0.7739 | 0.7682 | 0.7783 | 0.8976 | 0.8659 | 0.8928 |
| BART | 0.7380 | 0.7464 | 0.7574 | 0.8853 | 0.8585 | 0.9294 |
| BERT | 0.7488 | 0.7541 | 0.7651 | 0.8895 | 0.8653 | 0.9316 |
| COCA | 0.7769 | 0.7576 | 0.7703 | 0.8932 | 0.8623 | 0.9382 |
| ASE | **0.7884** | **0.7727** | **0.7839** | **0.8996** | **0.8701** | **0.9482** |

### Table 3: Performances of ablated models on AOL dataset.

| Metric | w/o. PFQ | w/o. PCD | w/o. PSQ | ASE |
|--------|---------|---------|---------|------|
| NDCG@1 | 0.4100 | 0.4036 | 0.4102 | 0.4144 |
| NDCG@3 | 0.5580 | 0.5570 | 0.5636 | 0.5682 |
| NDCG@5 | 0.5933 | 0.5895 | 0.5957 | 0.6007 |
| NDCG@10 | 0.6205 | 0.6180 | 0.6246 | 0.6283 |
| MAP | 0.5579 | 0.5546 | 0.5608 | 0.5650 |
| MRR | 0.5691 | 0.5650 | 0.5707 | 0.5752 |

### 5.2 Ablation Studies

To demonstrate the effectiveness of the generative tasks for helping the ranking task, we design several variants of ASE. Specifically, we conduct ablation experiments on AOL dataset as follows:

- **ASE w/o. PFQ.** We remove the task of Predicting Future Queries (PFQ).
- **ASE w/o. PCD.** We discard the task of Predicting future Clicked Documents (PCD).
- **ASE w/o. PSQ.** We abandon the task of Predicting a Supplemental Query (PSQ).

The performances are presented in Table 3. All the ablated models perform worse than the full ASE, which demonstrates the effectiveness of utilizing our generative tasks to model current search intent. Specifically, we can draw these conclusions:

1. **(1) Predicting future queries is effective for inferring the actual search intent.** In Section 3.4, we propose to treat future queries as a generation target because they have higher quality than the current one. After removing this task, ASE’s performance drops. Specifically, it drops about 1.80% in terms of NDCG@3. This demonstrates the effectiveness of this task.

2. **(2) Predicting future clicked documents can help the ranking task.** In Section 3.4, we propose to treat future clicked documents as another generation target because they are more accurate representations of search intent. After discarding this task, ASE’s performance decreases. Specifically, it decreases about 2.61% in terms of NDCG@1. This indicates that utilizing the information of future clicked documents can help the ranking task.

3. **(3) Predicting a supplemental query can make our model more robust.** In Section 3.4, we attempt to mine a supplemental query from other sessions to supplement our understanding of the current query. After abandoning this task, ASE’s performance declines. For example, it declines by about 1.01% in terms of NDCG@3. This shows that this task can make our model more robust.

### 5.3 Performances of Various Generative Targets

Given the current sequence \( S = \{q_1, d_1, q_2, d_2, \ldots, q_n\} \), we explore extensive possibilities of generation targets, e.g., preceding queries, the current query, subsequent queries, historical clicked documents, future clicked documents, and a supplemental query. We will treat each of them as the only generative task here and jointly learn to generate it with the ranking task. The performances of these generative targets on AOL dataset are presented in Table 4. Note that we only use one sequence per GT to get a straightforward view of its effectiveness. In the following section, we will study the length of GTS (i.e., the prediction window size \( w \)). From these results, we can draw the conclusions:
Table 4: Performances of different generative targets on AOL dataset. Suppose the current session sequence is $S = \{q_1, d_1, q_2, d_2, \ldots, q_n\}$.

| $GT$ | NDCG@$1$ | NDCG@$10$ | MAP |
|------|---------|----------|------|
| - (BART) | 0.3882 | 0.6124 | 0.5450 |
| $q_{n-1}$ | 0.3849 | -0.85% | 0.6103 | -0.34% | 0.5427 | -0.42% |
| $q_n$ | 0.3928 | +1.84% | 0.6077 | -0.77% | 0.5442 | -0.15% |
| $q_{n+1}$ | 0.4004 | +3.14% | 0.6150 | +0.42% | 0.5516 | +1.21% |
| $d_{n-1}$ | 0.3922 | +1.03% | 0.6104 | -0.33% | 0.5464 | +0.26% |
| $d_n$ | 0.4022 | +3.61% | 0.6212 | +1.44% | 0.5548 | +1.80% |
| $d_{n+1}$ | 0.4044 | +4.17% | 0.6206 | +1.34% | 0.5565 | +2.11% |
| $q_n$ | 0.3990 | +2.78% | 0.6151 | +0.44% | 0.5509 | +1.08% |

(1) Predicting the future sequences are more effective than simply recovering the historical ones. As explained in Section 3.4, the future queries and documents can represent the search intent. As presented in Table 4, we can notice that predicting the future behaviors achieves more considerable improvement over the ranking task than recovering the historical behaviors. Specifically, predicting the following query $q_{n+1}$ increases the performance of the ranking task by about 1.21% in terms of MAP on AOL, whereas recovering the previous query $q_{n-1}$ makes the performance drop by about 0.42% in terms of MAP.

(2) It is generally more helpful to predict the information of clicked documents than predict queries. As stated in Section 3.4, the clicked documents can often represent the search intent more accurately. As shown in Table 4, treating the clicked documents as generative targets perform better than queries. Specifically, predicting the current clicked document $d_n$, and the following query $q_{n+1}$ increase the performance of BART by about 3.61% and 3.14% in terms of NDCG@1 on AOL, respectively. We compare the results of $d_n$ and $q_{n+1}$ because they are considered the first behaviors in the subsequent sequences of $S$.

(3) Predicting a supplemental query can help the ranking task. As illustrated in Section 3.4, a supplemental query can supplement the understanding of the search intent. As shown in the last line of Table 4, predicting a supplemental query $q_n'$ increases the performance by about 1.08% in terms of MAP on AOL. This indicates that our third generative task can help the ranking task.

5.4 Effect of Prediction Window Size

In Section 3.4, we try to utilize the information of future sequences to help encode the current sequence. Specifically, we treat the future queries and clicked documents as generation targets in Task 1 and Task 2. We use a prediction window size $w$ to control the number of subsequent behaviors to generate. To determine the value of $w$, we finetune a variant of ASE (BART with $G\bar{T}_1$ and $G\bar{T}_2$) under different settings of $w$ on AOL dataset. We do not include Task 3 ($G\bar{T}_3$) in this variant because we want to directly estimate $w$’s influence on the ranking task without the effect of $G\bar{T}_3$. We find that our model performs best when $w$ is 2 on the validation sets. In Figure 4, we show the performances of this variant under different $w$ on the test set of AOL. Note that we show the results of the test set only for consistency with previous experiments’ results. $w$ is tuned based on performances on the validation sets.

From Figure 4, we can find the performance increases from 0 to 2 and slowly decreases from 2 to 4. We believe there is a trade-off. If $w$ is too small (0,1), the encoder can not actually encode the search intent into the high-level representations. And if $w$ is too large (3,4), the generation target may become too hard to generate. Besides, the average session lengths are about 2.5 for both datasets (Table 1), so there will be many empty sequences in the GTs if $w$ is too large.

5.5 Application to Other Transformer-based Encoder-Decoder Models

As illustrated in Section 3.2, we choose BART as our model’s backbone mainly for fair comparisons with BERT-based baselines (HBA, COCA). However, our approach can be easily applied to other Transformer-based encoder-decoder structured models. In this section, we choose two seq2seq models (T5 [27] and BlenderBot [23]) as the base models. For T5, we use the small version provided by Huggingface. For BlenderBot, we use the small version provided by Facebook on Huggingface. They are fine-tuned with the ranking task that is the same as BERT’s and BART’s strategy introduced
in Section 4.2. We also train them with our designed generative tasks, and the corresponding results are reported as “X+G Ts”.

As presented in Table 5, the models with our designed generative tasks outperform their base models significantly on AOL dataset, respectively. Specifically, T5-small model with GTs improves T5-small by more than 2.02% in terms of MAP. This indicates that utilizing our generative tasks to enhance session context modeling is effective under different backbones, and our approach can be easily applied to other Transformer-based encoder-decoder structured models than BART.

5.6 Performance on Different Query Positions and Sessions with Different Lengths

Following previous works [2, 6, 36, 37], in order to study ASE’s performance on sessions with different lengths, we split the test dataset of AOL as follows:

- Short sessions (with 2 queries) - 66.5% of the test set.
- Medium sessions (with 3-4 queries) - 27.24% of the test set.
- Long sessions (with 5+ queries) - 6.26% of the test set.

We compare ASE with Duet, HBA, and COCA on these different sessions. The results are presented in the left part of Figure 5. We can find that: (1) The ad-hoc ranking model Duet performs worse than context-aware models, which indicates the importance of modeling session context. (2) ASE outperforms other models on all lengths of sessions, which demonstrates the effectiveness of utilizing our generative tasks to model session context. (3) ASE performs worse on long sessions than on short sessions. As explained in [2, 36], long sessions are intrinsically more difficult. The similar declining trends of other models also demonstrate this idea.

To study ASE’s performance of modeling task progression, we also compare it with HBA and COCA on different query positions. The results are shown in the right part of Figure 5. We can find that the performances most increase as the session progresses because there is more session context to model. However, compared to COCA, ASE has a relatively slower speed for improvement (or performs better on queries that lack context). This is because ASE utilizes our generative tasks during training, which can help its encoder predict the search intent even with few historical behaviors. Besides, it is interesting that all models’ performances decrease from L4 to L7. We believe these long sessions often represent complex or exploratory search tasks, which are hard to complete.

6 Conclusions and Future Work

In this work, we attempt to utilize generative tasks to model session context. An encoder-decoder structure and three generative tasks are used to enhance the ability of the encoder. With these generative tasks, we aim to train our model to predict future queries, future clicked documents, and a supplemental query. We believe that if our model could predict these sequences, then the actual search intent has been successfully encoded into the high-level representations of the current session sequence. Rich experiments on two public search logs demonstrate the effectiveness and broad applicability of our approach.

Nevertheless, our work still has some limitations that we plan to address in future work: (1) Though ASE outperforms COCA without pre-training and data augmentation strategies, ASE still has potential variants that incorporate pre-training techniques. In future work, we will try to incorporate pre-training techniques (e.g., contrastive learning) and data augmentation strategies (e.g., curriculum learning) into ASE to further improve its performance. (2) The method of mining a supplemental query from the database is relatively naive. We plan to use more sophisticated algorithms or models (e.g., Sentence Transformer) to find a query with higher quality. (3) In order to further denoise the current session, the historical behaviors may be treated with distinction. For example, we could first extract or generate keywords from the behaviors before putting them into the encoder.

Acknowledgments
Zhicheng Dou is the corresponding author. Ji-Rong Wen is also with Key Laboratory of Data Engineering and Knowledge Engineering, MOE. This work was supported by the National Natural Science Foundation of China No. 61872370 and No. 61832017, Beijing Outstanding Young Scientist Program NO. BJJWZYJH01201910002009, and Intelligent Social Governance Platform, Major Innovation & Planning Interdisciplinary Platform for the “Double-First Class” Initiative, Renmin University of China.
of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017. ACM, 55–64. https://doi.org/10.1145/3077136.3080609

[35] Yujia Zhou, Zhicheng Dou, Yutao Zhu, and Ji-Rong Wen. 2021. PSSL: Self-supervised Learning for Personalized Search with Contrastive Sampling. In CIKM ’21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021, Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, and Hanghang Tong (Eds.). ACM, 2749–2758. https://doi.org/10.1145/3459637.3482379

[36] Yutao Zhu, Jian-Yun Nie, Zhicheng Dou, Zhengyi Ma, Xinyu Zhang, Pan Du, Xiaochen Zuo, and Hao Jiang. 2021. Contrastive Learning of User Behavior Sequence for Context-Aware Document Ranking. In CIKM ’21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021. ACM, 2780–2791. https://doi.org/10.1145/3459637.3482243

[37] Xiaochen Zuo, Zhicheng Dou, and Ji-Rong Wen. 2022. Improving Session Search by Modeling Multi-Granularity Historical Query Change. In WSDM ’22, The Fifteenth ACM International Conference on Web Search and Data Mining, February 21-25, 2022, Tempe, AZ, USA. ACM. https://doi.org/10.1145/3488560.3498415