GDESA: Greedy Diversity Encoder with Self-Attention for Search Results Diversification

XUBO QIN, School of Information, Renmin University of China, China
ZHICHENG DOU, Gaoling School of Artificial Intelligence, Renmin University of China, China
YUTAO ZHU, DIRO, University of Montreal, Canada
JI-RONG WEN, Beijing Key Laboratory of Big Data Management and Analysis Methods, Key Laboratory of Data Engineering and Knowledge Engineering, MOE, China

Search result diversification aims to generate diversified search results so as to meet the various information needs of users. Most of those existing diversification methods greedily select the optimal documents one-by-one comparing with the selected document sequences. Due to the fact that the information utilities of the candidate documents are not independent, a model based on greedy document selection may not produce the global optimal ranking results. To address this issue, some work proposes to model global document interactions regardless of whether a document is selected or not, which is inconsistent with actual user behavior. In this paper, we propose a new supervised diversification framework as an ensemble of global interaction and document selection. Based on a self-attention encoder-decoder structure and an RNN-based document selection component, the model can simultaneously leverage both the global interactions among all the documents and the interactions between the selected sequence and each unselected document. This framework is called Greedy Diversity Encoder with Self-Attention (GDESA). Experimental results show that GDESA outperforms previous methods that rely just on global interactions, and our further analysis demonstrates that using both global interactions and document selection is necessary and beneficial.

CCS Concepts: • **Information systems → Information retrieval diversity**.

Additional Key Words and Phrases: Search result diversification, Self-attention, Greedy selection

1 INTRODUCTION

According to studies, the majority of user-issued queries contain short, ambiguous, or unclear words or phrases [11, 14, 29, 30]. For example, when the query "Java" is issued, one user may search for information about "Java island", while another may want information about the "JAVA programming language". Even a single user may expect diverse results that can cover different aspects of their information need. Search result diversification is used to address this problem by returning a diversified document list that can satisfy a variety of information needs.

Existing models of search result diversification can be categorized into supervised and unsupervised, depending on whether supervised learning is applied. Most traditional approaches are unsupervised, and they rely on handcrafted features and functions [1, 5, 9, 13, 27]. In recent years, an increasing number of researchers have

Authors' addresses: Xubo Qin, School of Information, Renmin University of China, Beijing, China, qratosone@live.com; Zhicheng Dou, Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China, dou@ruc.edu.cn; Yutao Zhu, yutao.zhu@umontreal.ca, DIRO, University of Montreal, Montreal, Quebec, Canada; Ji-Rong Wen, Beijing Key Laboratory of Big Data Management and Analysis Methods, Key Laboratory of Data Engineering and Knowledge Engineering, MOE, Beijing, China, jirong.wen@gmail.com.

1046-8188/2022/6-ART $15.00
https://doi.org/10.1145/3544103

attempted to apply machine learning methods to search result diversification to automatically learn an efficient ranking function [34, 35, 41]. These supervised approaches model the diversity of each candidate document either by the subtopic coverage of the results [1, 9, 13, 15, 27] (also known as explicit approaches), or by the novelty based on document-document similarity without using subtopics [34, 35, 41, 45] (also known as implicit approaches).

Obtaining a global optimal ranking for search result diversification is a NP-hard problem, because the model has to search the entire ranking space by enumerating all possible permutations of documents. In general, there are three typical methods to tackle this problem:

(1) The first one is a greedy document selection strategy. This is the most common method that frames document ranking problem as a *greedy sequential document selection* – A diversification model fills each rank position sequentially by comparing each candidate document with the selected documents and selecting the most diversified one. The interaction between each candidate document and all selected documents can be used as diversity features. We refer to this type of interaction as **sequential interaction**. The greedy document selection strategy can be used as a simplification of the global optimal ranking and thus reducing the computing cost. Since users typically browse the document list from top to bottom, greedy document selection is also advantageous due to its resemblance to human behavior.

However, researchers [12] found that conventional greedy document selection approaches cannot always result in global optimal rankings. This is because previous methods focused exclusively on the interaction between each candidate document and the selected document sequence, neglecting the candidate document's interaction with other candidate documents. Because the information utilities of all candidate documents are interdependent, selecting one of them will influence the utilities of others. As a result, selecting each locally optimal document sequentially may not lead to a global optimal document ranking. This problem is even more severe when the selected sequence is short or empty (typically at the beginning stage of ranking). For instance, assuming there are three candidate documents $\{d_1, d_2, d_3\}$, where d_1 covers the subtopic q_1; d_2 covers q_2, q_3; d_3 covers q_1; and all three documents have similar relevance scores to the given query. At the beginning, the selected sequence is empty, so any of the candidate documents can be viewed as a "diversified document" for the empty selected sequence, namely the diversification scores of $\{d_1, d_2, d_3\}$ are indistinguishable. For such a case, a greedy selection-based diversification model will select d_1 for the first ranking position. However, because the diverse ranking task seeks to satisfy most user intents at the former position, d_2 is a better choice than d_1 in terms of intent-based diversification metrics (*e.g.*, α-nDCG). A straightforward solution for this problem is to introduce additional subtopic information, as some explicit approaches have done. However, in practice, it is difficult to collect the ground truth of user intent coverage for each document in ranking tasks. As an alternative, existing explicit diversification models employ automatically mined subtopics, which are different from the actual user intents. Consequently, these models cannot estimate the user intent coverage of each document accurately. In other words, explicit diversification methods cannot solve such a "cold start" problem completely.

(2) Another possible solution is to consider the entire candidate document sequence as a whole and model the interactions between all candidate documents globally. We refer to this kind of document interaction as **global interaction**. In recent years, several multivariate ranking models have been proposed [2, 20]. They use sequential neural networks (*e.g.*, LSTM or Transformer [33]) to learn the global interactions between all candidate documents. The documents are ranked according to the ranking scores computed by the model. Nevertheless, such kind of methods ignores the sequential interactions and pays less attention to the document selection process. Indeed, in search result diversification, the selected documents and candidate ones play different roles in the process. The next selection is highly relied on the previous selection status. The global interaction based methods is hard to accurately rank all documents in one time. Moreover, things are more severe when some documents are ranked at the sub-optimal positions. The sequential interaction-based methods can adjust the

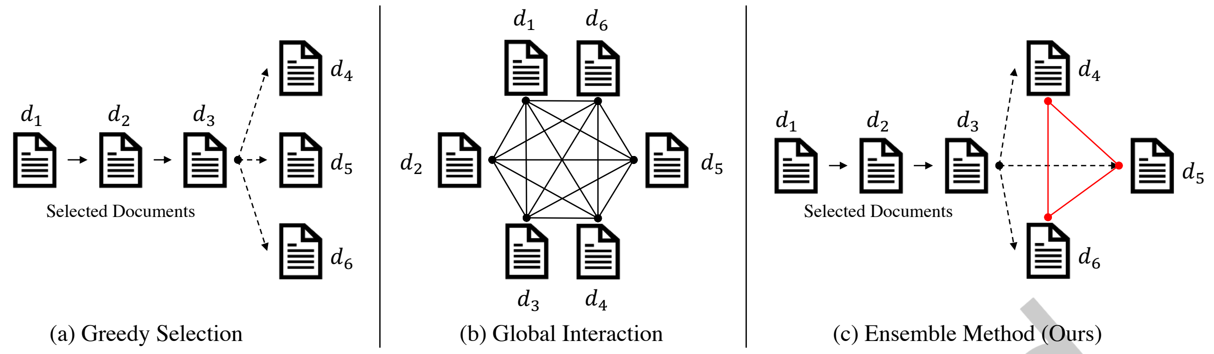

Fig. 1. The comparison of sequential interaction and global interaction. For the sequential selection models, each candidate document can interact with the selected document sequence, but the candidate documents themselves are independent of each other. For models that consider global interaction (*e.g.*, DESA), all documents can interact with each other, and the model does not distinguish whether a document is selected or not. Our proposed GDESA is an ensemble framework, in which both global interaction and sequential interaction are taken into account.

latter part of the sequence so as to reduce the redundancy in previous part, but the global interaction based methods are helpless on these cases.

(3) In addition to applying greedy search or considering global interaction, researchers also tried Monte-Caro Tree Search (MCTS) to explore a larger ranking space and increase the probability of selecting the global optimal document [12]. However, as a deep reinforced learning model, it is difficult to train and deploy in real applications since MCTS is extremely time-consuming. Besides, since MCTS is a sequential process that is incompatible with parallel computing, it is difficult to accelerate the model training with modern powerful hardware such as GPUs and other parallel computing devices.

In this paper, we propose a new search result diversification framework that combines both global interaction and sequential interaction. We call this framework **G**reedy **D**iversity **E**ncoder with **S**elf-**A**ttention (GDESA). The comparison between our framework and existing methods is shown in Figure 1. **For global interaction**, we employ a self-attention based encoder-decoder structure (similar to our previous work [22]) to model the global document interactions between candidate documents and subtopics. In the framework of GDESA, we use a self-attention based encoder to model the candidate document sequence and the subtopics. This encoder component can measure the global document interactions between each document in the candidate sequence for re-ranking, while the global interactions indicate the novelty of each candidate document. Additionally, our framework contains an optional decoder component that can learn the subtopic coverage of different documents when the subtopics are available. We also analyze the mechanism how self-attention works in diversification tasks theoretically. **For sequential interaction**, we equip our framework with an RNN-based greedy sequential selection component. It can greedily select the best candidate document at each ranking position. By this means, our method can inherit the advantages of the greedy document selection strategy. Instead of applying reinforcement learning and MCTS, our framework is based on multi-head self-attention and supervised learning. The training/inference can be well-supported by parallel computing hardware.

Experimental results on the TREC Web Track data show that our GDESA model outperforms the state-of-the-art implicit and explicit diversification models. The improvement of GDESA over DESA demonstrates the effectiveness of our proposed greedy sequential selection component and the combination of global and sequential interaction. Moreover, our detailed analysis indicates that introducing the new component has less influence on training and inference efficiency.

The contributions of this paper are summarized as follows:

(1) We propose a search result diversification framework that incorporates both global document interaction and sequential interaction. Compared with the previous approaches that are solely based on greedy sequential selection, our framework has a higher probability of achieving the global optimal ranking. Additionally, our framework is more similar to the actual user behavior in diversification tasks than global interaction based approaches.
(2) More specifically, we use a self-attention based encoder-decoder structure for modeling global document selections. We then theoretically analyze why self-attention is suited to the task of search result diversification. Besides, we employ an RNN-based component to model the selected sequence in order to select candidate documents greedily.
(3) Experimental results demonstrate that our proposed framework can outperform existing implicit and explicit approaches significantly. Both training and inference are effective and efficient. Subsequent experiments and analysis indicate the value of leveraging both global interaction among all the documents and sequential interaction between selected document sequence and candidate documents.

This paper is an extension of our previous work [22] which has been presented at the Conference on Information and Knowledge Management (CIKM 2020). The main extensions of this paper are listed as follows:

(1) Comparing with our previous framework of DESA, the new GDESA framework can leverage both global and sequential document interactions. We further demonstrated that it's necessary and beneficial to incorporate the two kinds of interactions to improve the performance of search result diversification.
(2) We measure the training and inference time of DESA and GDESA, demonstrating that GDESA is effective and efficient compared with the previous MCTS-based work [12].
(3) We extend the experimental results with new baselines including DALETOR [39], DVGAN [18], MDP-DIV [36] and PPG-DIV [38]. Based on the results of those new baselines, we also analyze the effect of initial ranking lists in diversification tasks.
(4) We propose a discussion on relevance and diversity, and we further demonstrated that our proposed GDESA framework is robust to deal with poor initial ranking lists[1].

The rest of the paper is organized as follows. In Section 2, we introduce some related work, including a brief introduction of search result diversification, the sequential-based approaches, and the global-based approaches. In Section 3, we introduce the structure of our GDESA framework and explain how the framework leverages both global interaction and greedy document selection. In Section 4, we describe the self-attention based components in detail. Section 5 contains the description of the sampling and optimization process, the analysis of how self-attention works in search result diversification task, the analysis of time complexity, and the comparison between our GDESA framework and previous methods. In Section 6, we report our experimental result, and we analyze it in detail. Finally, we make a conclusion of our work in Section 7.

2 RELATED WORK

2.1 Background of Search Result Diversification

Most of the traditional ranking models in Information Retrieval are based the Probability Ranking Principle (PRP) [25] hypothesis, which assumes that all the documents are independent one another. While in search result diversification, a document's diversity is based on its novelty compared with others, so the model has to consider the dependencies between different documents.

[1]In our paper, "poor" means that there are only a few intent-covered documents in initial ranking lists, while "rich" means that there are many candidate documents with intents covered. In the view of evaluation metrics, "intent-covered" means that this document has got a positive diversify judgment.

Table 1. Categorization of previous sequential-based diversification approaches.

	Unsupervised	Supervised	Reinforced
Explicit/Integrated	IA-Select, HxQuAD, PM2, TPM2, TxQuAD, xQuAD, HPM2	DSSA, DVGAN	-
Implicit	MMR	SVM-DIV, R-LTR, PAMM, NTN	MDP-DIV, PPG-DIV, M2DIV

Existing diversification approaches can be classified as implicit or explicit, depending on whether user intents (represented as subtopics) are explicitly modeled. The implicit diversification approaches measure the similarity between each candidate document and the previously selected documents and assumes that a novel candidate document should be dissimilar with the selected documents. The most typical implicit model is the Max Margin Relevance (MMR) [5] model:

$$\text{Score}_{\text{MMR}} = \lambda \text{score}(d_i, q) - (1 - \lambda) \max_{d_j \in S} \text{sim}(d_i, d_j), \tag{1}$$

where $\text{score}(d_i, q)$ denotes the relevance score of the current document candidate d_i and the given query q, and $\text{sim}(d_i, d_j)$ denotes the similarity of d_i to the selected document d_j in the selected set S. The less similar the candidate document is to the selected documents, the more diverse it will be. The final ranking score of the candidate document is a linear combination of the relevance and novelty scores.

The explicit approaches measure the diversity of documents by explicitly modeling the intent coverage of documents. Those user intents are represented as subtopics. In explicit diverse ranking, a candidate document is considered diverse if it covers as many new subtopics for the given query as possible when compared with the selected document sequence. Ideally, explicit approaches would outperform implicit approaches since they can explicitly model subtopic coverage. While in practice, online ranking tasks lack the ground truth for intent coverage, and the mined subtopics may not be comparable to the actual user intents. As a result, the explicit approaches may not always outperform than implicit methods.

Besides, according to a recent literature [19], existing methods can also be divided into approaches with sequential interaction and approaches with global interaction. In the following section, we will introduce them in detail.

2.2 Sequential-based Diversification Approaches

Most existing approaches are based on sequential document selection to generate the diversified ranking lists. These approaches are referred to as “sequential-based approaches”. They usually compare each candidate document with the selected document sequence, then select and append the next document to the selected document sequence. This process is repeated until all candidate documents are selected. Some sequential-based methods are implicit approaches [23, 31, 34, 35, 41, 45]), while others are explicit approaches [9, 13, 15, 27]. Recent years, integrated approaches have also been proposed to integrate both implicit and explicit features for diversification. DVGAN [18] is a typical method that uses an implicit model and an explicit model as the generator and discriminator respectively in a GAN (Generative Adversarial Networks) structure. According to the optimization process, existing sequential-based methods can also be divide into unsupervised approaches, supervised approaches, and reinforced approaches, as shown in Table 1.

All sequential-based methods apply a greedy selection strategy. They independently compare each single candidate document to the selected document sequence and fill in the document ranking list with the locally optimal document one by one. Interaction between all the candidate documents is neglected. Since the actual information utilities of the candidate documents are not independent, this strategy cannot guarantee a global optimal ranking. Some researchers [12, 36] proposed to use Monte-Caro Tree Search (MCTS) to search a larger ranking space and minimize the gap between local optimal and global optimal rankings. However, the method is difficult to train due to its high time requirement, and it can only model the document novelty but neglects the subtopic coverage.

2.3 Global-based Learning-to-Rank Approaches

For ad-hoc ranking task, it has been reported that global inter-document interaction is beneficial for improving the ranking performance in online systems [2, 24]. Based on this observation, researchers have proposed various approaches, including DLCM [2], SetRank [20], and DIN [21]. These approaches are based on multivariate scoring functions that are implemented using a recurrent neural network or multi-head self-attention network. They take the entire document sequence as input and simultaneously return all ranking scores. The final ranking list is produced by sorting all the documents according to their ranking scores. These approaches are referred to as "global-based approaches", because they measure the interactions between all candidate documents on a global scale.

As an early exploration, DLCM [2] leveraged an LSTM network with an attention mechanism to model the document sequence. Later, since a self-attention mechanism (also known as self-attention network) has achieved great success in many NLP tasks [10, 33, 43, 46], researchers developed various ranking models using self-attention networks to better model document interaction. Typical methods, such as DIN (Document Interaction Network) [21] and SetRank [20], use a Transformer-like [33] multi-head self-attention encoder. Additionally, researchers found that the self-attention network can provide a permutation equivalent for the document list, which aids in ranking task.

Given that the sequential-based diversification approaches overlook interaction between candidate documents, it is straightforward to extend the global-based learning-to-rank approaches for diversification. For example, DALETOR [21] is a global-based diversification approach based on DIN, and DESA [22] is based on SetRank. However, those global-based ranking approaches do not distinguish whether a document is selected or not, so the sequential selection process is neglected. Indeed, the selected documents and candidate documents play different roles in search result diversification. The selection process is relied on the status of previous selected document list. Besides, it is common for a user to browse documents from top to bottom, so the top results are much more important for diversification. Therefore, we believe sequential interaction is still beneficial for diversification task.

2.4 Other Diversification Approaches

There are a few other diversification approaches. For example, Yigit-Sert et al. [40] proposed three approaches to transform the subtopic matching and weighting problem to a learning-to-rank problem. The methods depend on extra features specialized for explicit diversification based on known query aspects. Liang et al. [17] proposed a supervised learning framework for enhancing the diversification performance with personalized diversification. Different from these methods, we focus on search result diversification without additional information.

3 PROPOSED METHOD

In this section, we will give the problem statement of search result diversification and introduce the overall framework of our proposed GDESA. More details about the implementation and optimization will be described in Section 4 and Section 5.

Table 2. Notations used in this paper.

Notation	Description
q	Input query
$\mathcal{I}$,q_i	All subtopics corresponding to q
q_i	A single subtopic, $q_i \in \mathcal{I}$
$\mathcal{D}$	Candidate document sequence
$\boldsymbol{D}$	Embeddings of candidate document sequence
$\boldsymbol{I}$	Embeddings of all subtopics
$\mathcal{R}$	Returned document rank list
$x_{d,q}$	Relevance features of document d to query q
x_{d,q_i}	Relevance features of document d to subtopic q_i
$\mathbf{d}$	Initial document embedding for the document d
$\boldsymbol{q}_i$	Initial subtopic embedding for the subtopic q_i
$\boldsymbol{h}_d^{\text{enc}}$	Encoder output for the document d
$\boldsymbol{h}_d^{\text{dec}}$	Decoder output for the document d
$\mathbf{h}_d^{\text{DS}}$	Output of document selection component for the document d
$\mathbf{h}_C^{\text{DS}}$	Hidden state of selected context C
s_{q_i}	Relevance score of the document to the subtopic q_i
$\mathcal{S}_{d,\mathcal{I}}$	Combination of all the subtopic satisfaction scores for d
$\boldsymbol{v}_{d,q,\mathcal{I}}$	Document vector for generating the ranking score
$[;]$	Concatenation operation

3.1 Problem Formulation

Table 2 lists the notations used in this paper and their descriptions. For the query q, the corresponding subtopics are represented as $\mathcal{I}$, where $|\mathcal{I}| = k$. And q_i is the i-th subtopic ($i \in [1, k]$ and $q_i \in \mathcal{I}$). Given q and the initial ranking list $\mathcal{D}$ which includes a group of candidate documents, the diversification approach aims at re-ranking $\mathcal{D}$ to generate a diversified ranking list $\mathcal{R}$.

Different from the ad-hoc retrieval task which focuses on returning relevant documents, search result diversification must taken into account both the relevance of each individual document and its novelty compared with the selected document sequence. As we described in Section 1, most existing diversification models are based on the greedy selection approach: each of the documents in the diversified lists is iteratively selected by evaluating the relevance of each remaining document and the novelty it adds to the previous selected documents. It must be noted that in those previous models, each candidate document is compared separately to the selected document sequence. The interaction between all candidate documents is overlooked, which may lead to suboptimal ranking results.

Our method is based on DESA, in which all diverse ranking scores for each candidate document are computed simultaneously, and the diverse ranking list is generated by directly sorting the documents according to their scores, without resorting to greedy document selection. However, as discussed in Section 2.3, the greedy document selection process is also beneficial for diverse ranking. Due to the permutation equivalence of self-attention[20],

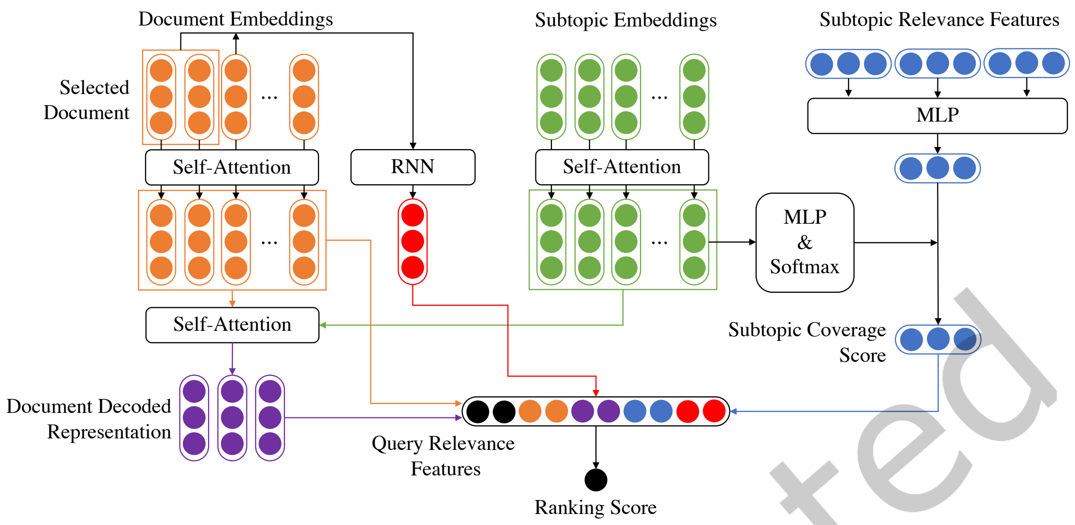

Fig. 2. The overall structure of GDESA. The framework takes the entire candidate document sequence and subtopics together as input, and returns the encoded and decoded representations of each candidate document simultaneously. For the t-th document d_t, the static features of the document includes query relevance features $\mathbf{x}_q$, the encoded and decoded representation $\mathbf{h}_t^{\mathrm{enc}}$ and $\mathbf{h}_t^{\mathrm{dec}}$, and subtopic coverage scores $\mathcal{S}_{d,\mathcal{I}}$, while the dynamic features $\mathbf{h}_d^{\mathrm{DS}}$ are based on the the hidden state of selected sequence $\mathbf{h}_C^{\mathrm{DS}}$. The ranking score s_t is obtained by the combination of static and dynamic features with a learning-to-rank function.

the DESA framework without greedy document selection may fail to accurately measure the effect of ranking positions.

To tackle this problem, we propose a GDESA framework that combines both global interaction and greedy document selection process.

3.2 Overview

Figure 2 shows the overall structure of our proposed GDESA framework. Similar to existing document selection-based approaches, GDESA greedily selects the next best document and sequentially generates $\mathcal{R}$. The difference is that the candidate documents in GDESA can interact with each other globally before the ranking starts. For each ranking position, the ranking score s for the candidate document d can be described as:

$$s = \mathrm{GDESA}(\mathcal{D}, q, \mathcal{I}, C, d), \tag{2}$$

where C is the context state of the previous selected document sequence. The features used in GDESA are a combination of static and dynamic features, which can be described as follows:

$$\mathrm{GDESA}(d) = \mathrm{LTR}\left(\left[\underline{\mathrm{Stat}\left(\mathcal{D}, q, d, \mathcal{I}\right)}; \underline{\mathrm{Dyn}(C, d)}\right]\right). \tag{3}$$

Here GDESA(d) is the shortened form of GDESA$(\mathcal{D}, q, \mathcal{I}, C, d)$. Stat$(\cdot)$ denotes the static features, while Dyn$(\cdot)$ denotes the dynamic features. LTR denotes a learning-to-rank function, and $[;]$ is the concatenation operation. The static features represent the properties of each document in initial sequence regardless of whether a document is selected or not. Conversely, the dynamic features are changing during the progression of the greedy document selection process. At each ranking position, the document with the best ranking score will be selected and appended to the selected document sequence. The two components can be trained in an end-to-end manner.

3.3 Dynamic Feature Component

The dynamic features are changing along with the greedy document selection process. We build a document selection component to model the sequential interaction features and select the most diversified candidate documents at each position depending on the previous selected document sequence.

According to previous research, a self-attention network is permutation equivalent [20], which means that it is ineffective at modeling the sequential interaction between a selected document sequence and candidate documents. Inspired by previous work [15], we apply an RNN-based model as the document selection component and use the unsupervised method doc2vec [16] to generate the initial document representations rather than building the optimized document representations automatically. For the candidate document d, the component based on RNN cell at the t-th ranking position can be described as:

$$\mathbf{h}_d^{\mathrm{DS}} = \tanh(\mathbf{W}_n[\mathbf{h}_C^{\mathrm{DS}}; \mathbf{d}] + \mathbf{b}_n), \tag{4}$$

where $\mathbf{W}_n$ and $\mathbf{b}_n$ are parameters. The RNN cell takes previous hidden states $\mathbf{h}_C^{\mathrm{DS}}$ and a candidate document $\mathbf{d}$ as input, and returns the next hidden state $\mathbf{h}_d^{\mathrm{DS}}$. In our implementation, we apply LSTM as the RNN cell due to its superiority on modeling long sequence. Consequently, $\mathbf{h}_d^{\mathrm{DS}}$ is used as the dynamic features:

$$\mathrm{Dyn}(C, d) = \mathbf{h}_d^{\mathrm{DS}}. \tag{5}$$

3.4 Static Feature Component

The static features Stat $(\mathcal{D}, q, d)$ of the document d in GDESA are generated by the component of global document interactions, which can be seen as a variation of DESA framework. Specifically, the static features of the candidate document d are:

$$\mathrm{Stat}(\mathcal{D}, q, d, \mathcal{I}) = [\mathbf{x}_{d,q}; \mathbf{h}_d^{\mathrm{enc}}; \mathbf{h}_d^{\mathrm{dec}}; \mathcal{S}_{d,\mathcal{I}}]. \tag{6}$$

Here $\mathcal{S}_{d,\mathcal{I}}$ is the combination of all subtopic satisfaction scores for d, $\mathbf{x}_{d,q}$ is the combination of relevance features between d and q. $\mathbf{h}_d^{\mathrm{enc}}$ and $\mathbf{h}_d^{\mathrm{dec}}$ are document representations of d generated by self-attention encoder and decoder. The static feature component of GDESA framework takes the initial ranking as input, and returns the static features including the document representations, relevance features, and subtopic coverage. These features are static, which means that they are not influenced by the document selection process. More details are provided in Section 4.

3.5 Ranking Process

As described in Equation 3, a learning-to-rank function takes the combination of static and dynamic features for each candidate document d as input, and returns its ranking score s as follows:

$$s = \tanh(\mathbf{w}_v^\top \mathbf{v} + \boldsymbol{b}), \tag{7}$$

$$\mathbf{v} = [\mathbf{h}_d^{\mathrm{enc}}; \mathbf{h}_d^{\mathrm{enc}}; \mathbf{x}_{d,q}; \mathcal{S}_{d,\mathcal{I}}; \mathbf{h}_d^{\mathrm{DS}}]. \tag{8}$$

Most previous models are solely based on greedy sequential selection [28]. These models compare each candidate document with the selected sequence, and the interaction between other candidate documents are neglected. On the contrary, DESA takes a non-diversified initial ranking sequence as input, models the global interaction, and simultaneously returns the diversified ranking scores of all documents. The ranking list is generated by sorting all the candidate documents with their ranking scores without a document selection process.

Algorithm 1 Greedy Selection Ranking of GDESA

```
Input:
  The initial document sequence for ranking: $D = \{d_1, \ldots, d_n\}$
  The corresponding query: $q$
Output:
  The diversified document sequence $R$
  $R \leftarrow \emptyset$
  $\mathbf{H}_D^{\text{global}}$ = SelfAttn($D$) // pre-compute the global document interaction representations with the Self-Attention encoder and dencoder
  $C = \mathbf{h}_0^{DS}$ // initialize the first hidden state for the empty $|R|$
  while $|R| < |D|$ do
    $S_D = \text{GDESA}(\mathbf{H}_D^{\text{global}}, q, C)$
    $d = \arg\max(S_D)$
    $R \leftarrow R \bigcup \{d\}$
    $C \leftarrow \mathbf{h}_d^{DS}$ //update $C$ with the hidden state corresponding to the selected document
  end while
  return $R$
```

In our GDESA, both the document selection process and global interaction between documents are taken into account. In general, GDESA takes all interaction between selected and unselected documents as ensemble features. In ranking phase, it first takes the initial ranking sequence $\mathcal{D}$, query q and subtopics $\mathcal{I}$ as input, and returns the static features $\text{Stat}(\mathcal{D}, q, \mathcal{I})$ for the entire sequence $\mathcal{D}$. Since the static features of the documents are independent with the document selection process, $\text{Stat}(\mathcal{D}, q, \mathcal{I})$ can be pre-computed at the beginning of the ranking process, thus reducing the inference latency of the framework. When the static features of the documents are computed, the framework focuses on the dynamic features for document selection.

For each ranking position t, the dynamic feature component is initialized with the hidden state $\mathbf{h}_C^{\text{DS}}$ corresponding to the selected document sequence context C. When the component is initialized, $\mathbf{h}_d^{\text{DS}}$ is computed for each candidate document d, and the ranking score s is calculated by the combination of $\mathbf{h}_d^{DS}$, $\mathbf{x}_{d,q}$,$\mathcal{S}_{d,\mathcal{I}}$ and the pre-computed $[\mathbf{h}_d^{\text{enc}}, \mathbf{h}_d^{\text{dec}}]$. Once all candidate documents have been scored, the model can greedily select the best candidate document with the highest score and append it to the selected document sequence. When the document d is selected, $\mathbf{h}_C^{\text{DS}}$ will be updated with $\mathbf{h}_d^{\text{DS}}$ and the document selection component will be re-initialized with the updated $\mathbf{h}_C^{\text{DS}}$. This process is summarized in Algorithm 1. As a result, the state of the document selection component is updated according to the document selection process, while the document representations for global interaction remain unchanged. Due to the fact that the self-attention encoder is computed only once, the computational cost of the GDESA framework will not be prohibitively high for online ranking tasks.

4 SELF-ATTENTION BASED STATIC FEATURE COMPONENT

In this section we will introduce the static feature component of GDESA based on self-attention networks. This component is a combination of relevance features, subtopic coverage scores, and self-attention based document representations.

4.1 Overall Structure

Similar to DESA, the static feature component takes the entire candidate document sequence as input and models the interaction between all candidate documents by measuring their information utilities globally. More specifically, we inherit the encoder-decoder structure based on self-attention networks in DESA to model the relationship between each document in $\mathcal{D}$ and each subtopic $q_i \in \mathcal{I}$. The encoder component takes the entire candidate document sequence $\mathcal{D}$ as input, and returns the representations of all documents simultaneously. Since the self-attention encoder allows the documents to interact with each other globally, those document representations can reflect the novelty or dissimilarity of a document. After the encoder-based representations of both document sequence and subtopics are produced, the decoder component will take those two kinds of representations as input, returning the decoded document representations indicating the subtopic coverage of the documents. This framework is flexible and the decoder component is optional. It can be removed when subtopics are not available. Under this circumstance, the component will work implicitly by modeling the document interaction only.

4.1.1 Document Representations. For the static feature component, we use the same initial document representations as the ones used in dynamic feature component. We use the unsupervised method doc2vec [16] to generate the preliminary document representations instead of building the representations automatically.

4.1.2 Self-attention Encoder. The self-attention encoder in the static feature component takes D as input and returns the representations $\mathbf{H}_D^{\text{enc}}$ of the entire document sequence. When the subtopics are available, the encoder also takes the embeddings of subtopics I as input and returns the representations $\boldsymbol{H}_I^{\text{enc}}$ for all subtopics, *i.e.*, we have:

$$\mathbf{H}_D^{\text{enc}} = \text{SelfAttnEnc}(D), \quad \mathbf{H}_I^{\text{enc}} = \text{SelfAttnEnc}(I), \tag{9}$$

where the self-attention encoder is denoted as SelfAttnEnc$(\cdot)$. In the next section we will describe the decoder in details.

4.1.3 Self-attention Decoder. The decoder takes the encoded representation of document sequence $\mathbf{H}_D^{\text{enc}}$ and subtopics $\mathbf{H}_I^{\text{enc}}$ as input, and returns the decoded representations $\mathbf{H}_D^{\text{dec}}$ for all the documents. For every document, the decoded representation indicates the document's subtopic coverage. This step can be described as the following equations, where the decoder is denoted as SelfAttnDec$(\cdot)$:

$$\mathbf{H}_D^{\text{dec}} = \text{SelfAttnDec}(\mathbf{H}_D^{\text{enc}}, \mathbf{H}_I^{\text{enc}}), \quad \mathbf{h}_d^{\text{enc}} = \mathbf{H}_D^{\text{enc}}[\text{index}(d)], \quad \mathbf{h}_d^{\text{dec}} = \mathbf{H}_D^{\text{dec}}[\text{index}(d)], \tag{10}$$

where $\mathbf{H}_D^{\text{enc}}[\text{index}(d)]$ denotes the vector at index d in $\mathbf{H}_D^{\text{enc}}$. For the document d, the encoded and decoded representations $\mathbf{h}_d^{\text{enc}}$ and $\mathbf{h}_d^{\text{dec}}$ are used to obtain the document's ranking score. The component of modeling subtopic coverage is called as "decoder" in our paper. While in some other works [32, 44, 47–49], this operation is also known as "cross-attention".

4.1.4 Subtopic Coverage Scores. For each subtopic $q_i \in \mathcal{I}$, $(i \in [1, |\mathcal{I}|])$, we use a linear learning-to-rank function to learn the coverage score s_{d,q_i} through the subtopic relevance features $\mathbf{x}_{d,q_i}$:

$$\mathcal{S}_{d,\mathcal{I}} = [s_{d,q_1}; \ldots; s_{d,q_{|\mathcal{I}|}}], \quad s_{d,q_i} = \mathbf{x}_{d,q_i}^{\top}\mathbf{w}_r \cdot w_i.$$

Here $\mathbf{w}_r$ is a parameter, and w_i is the weight of the subtopic q_i. Following previous work [15, 22, 31], we use 18 traditional IR features for $\mathbf{x}_{d,q}$ and $\mathbf{x}_{d,q_i}$. Those features are listed as follows:

(1) BM25, TF-IDF and LMIR scores. Each of those three features is applied to those following levels: body, title, anchor, URL and the whole passage.
(2) The PageRank score of the document.
(3) The number of incoming and outgoing links.

To enhance the subtopic coverage scores and reduce the effect of latent subtopic redundancy, we reuse the subtopic representations of $\mathbf{H}_I^{\text{enc}}$ to obtain a score for each subtopic, which is then converted into a subtopic weight using a softmax function:

$$\mathbf{S}_I = (\mathbf{H}_I)^\top \mathbf{w}_\mathrm{I}, \quad \mathbf{W}_I = \mathrm{softmax}(\mathbf{S}_I), \quad w_i = \mathbf{W}_I[\mathrm{index}(q_i)]. \tag{11}$$

These scores denote the encoder-based weights of the subtopics. When a subtopic is redundant, its weights will be penalized by the self-attention encoder. Details will be described in Section 4.3.

4.1.5 *Summary of Static Features.*

The summarized document feature vectors $\mathbf{v}_{d,q,\mathcal{I}}^\top$ are concatenated by the following components: the query relevance features $\mathbf{x}_{d,q}$, the encoded document representation $\mathbf{h}_d^{\mathrm{enc}}$ and decoded document representation $\mathbf{h}_d^{\mathrm{dec}}$, and the coverage scores of all the subtopics $\mathcal{S}_{d,\mathcal{I}}$. Note that we use the same set of ranking features for query q as those used for subtopics. This feature vector $\mathbf{v}_{d,q,\mathcal{I}}^\top$ can be used as the output of static feature component:

$$\mathrm{Stat}(\mathcal{D}, q, d, \mathcal{I}) = \mathbf{v}_{d,q,\mathcal{I}}^\top, \quad \mathbf{v}_{d,q,\mathcal{I}}^\top = [\mathbf{x}_{d,q}; \mathbf{h}_d^{\mathrm{enc}}; \mathbf{h}_d^{\mathrm{dec}}; \mathcal{S}_{d,\mathcal{I}}]. \tag{12}$$

In the remaining part of this section, we will describe the implementation of self-attention based encoder and decoder.

4.2 Self-Attention Encoder

In GDESA, the self-attention encoder component denoted as SelfAttnEnc(·) can measure the global interactions between all the items in the given input sequence. It takes the whole candidate document sequence $\mathbf{D}$ as input, returning all the hidden states $\mathbf{H}_D^{\mathrm{enc}}$ simultaneously. These hidden states are used as the document representations with global interactions, which can indidate the novelty of every candidate document comparing with the other ones. In this section we will introduce the implementation of self-attention encoder in detail.

4.2.1 *Attention Function.*

In diversification task, the self-attention layers take the document embeddings as input. Different from RNNs, a self-attention network does not model the sequence information explicitly, so the standard Transformer structure also includes an optional component of positional encoding to incorporate the sequence information. In our work, we use trainable position embeddings which is the same as BERT [10]. These embeddings can capture the sequence information of the documents by adding them to the input document embeddings.

The self-attention component is implemented with multi-layer Transformer encoder blocks, which are based on the scaled dot-product attention function denoted as Attn(·), as follows:

$$\mathrm{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}})\mathbf{V}. \tag{13}$$

where $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ denote the query[2], key and value matrices of the attention function, respectively. In diversification tasks, the model takes the sequence of document representations $\mathbf{D}$ as input, and the query matrix can be defined as $\mathbf{Q} = \mathbf{D}$.

4.2.2 *Multi-Head Attention.*

Following previous work [20], we use the multi-head strategy to learn multiple aspects of different documents. The multi-head attention strategy, denoted as MultiHead(·), first projects the inputs $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ into h different heads with the dimension $\hat{E} = E/h$:

$$\mathrm{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = [\mathbf{a}_1; \ldots; \mathbf{a}_h], \quad \mathbf{a}_i = \mathrm{Attn}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V), \quad i \in [1, h]. \tag{14}$$

Here, all the $\mathbf{W}$s are parameters. Previous research [20] has shown that using the multi-head strategy can help the self-attention network to learn better document similarity distribution concerning multi-aspects. For the self-attention, we have $\mathbf{Q} = \mathbf{K} = \mathbf{V}$.

[2]Here "query" represents the query in dot-product attention, not the "query" in information retrieval.

4.2.3 Overall Structure of Self-Attention Encoder. The overall structure of the encoder component is a multi-layer stack of multi-head self-attention blocks. Similar to the original Transformer encoder block, each of these self-attention encoder layers contains a dropout layer and a fully connected feed-forward network (denoted as FeedForward(·)) with ReLU function as activation function. The i-th layer of the block is denoted as MSB_i and the encoder component SelfAttnEnc(·) with L layers can be described as follows:

$$\mathrm{SelfAttnEnc}(\mathbf{D}) = \mathrm{MSB_L}(\mathrm{MSB_{L-1}}(\ldots \mathrm{MSB_1}(\mathbf{D})), \tag{15}$$

$$\mathrm{MSB}(\mathbf{H}_{\mathrm{prev}}) = \mathrm{LayerNorm}(\mathbf{X} + \mathrm{FeedForward}(\mathbf{X})), \tag{16}$$

$$\mathbf{X} = \mathrm{LayerNorm}(\mathbf{H}_{\mathrm{prev}} + \mathrm{MultiHead}(\mathbf{H}_{\mathrm{prev}}, \mathbf{H}_{\mathrm{prev}}, \mathbf{H}_{\mathrm{prev}})), \tag{17}$$

where LayerNorm(·) denotes the layer normalization operation [3], and $\mathbf{H}_{\mathrm{prev}}$ is the output hidden state matrix of the previous encoder layer. For the first layer, $\mathbf{H}_{\mathrm{prev}} = \mathbf{D}$.

After multi-layers of multi-head attention interactions, the output hidden states of n input documents $\mathbf{H}_{\mathrm{output}} = [\mathbf{h}_1^{\mathrm{enc}}, \ldots, \mathbf{h}_n^{\mathrm{enc}}]$ can be used as the encoded document representations $\mathbf{H}_D^{\mathrm{enc}}$. This representation can indicate the novelty of a document, and the learning-to-rank function can use this representation to judge whether a candidate document is novel or redundant compared with other candidate documents.

4.3 Self-Attention Decoder

As described in Section 4.1.2, the encoder can also take the subtopics as inputs, and return the encoded representations of the subtopics. This is because the subtopic embeddings we used are actually the document embeddings. We used the subtopic embeddings released by Jiang et al. [15] based on the doc2vec representations of pseudo documents. More details about those subtopic embeddings can be found in Jiang's paper.

The encoded subtopic representations are important to the decoder since these representations include the attention distributions of the subtopics. In diverse ranking tasks, the available subtopics are mined from the query, and they are usually more than the actual user intents. Comparing with the user intents, the subtopics may still contain redundancy and mislead the diversification model. As the encoded subtopic representations include the encoder attention distributions of the subtopics, these distributions can be used to leverage the subtopics' potential redundancy and minimize the misleading effect. The effort of redundancy reducing is identical to the encoder-based subtopic weights mentioned in Section 4.1.4.

The decoder structure takes the representation of documents as query matrix, and subtopics as key and value matrix, returning the $\mathbf{H}_{\mathrm{dec}}$ representation matrix for the documents with the multi-head attention function:

$$\mathbf{H}_{\mathrm{dec}} = \mathrm{SelfAttnDec}(\mathbf{H}_D^{\mathrm{enc}}, \mathbf{H}_I^{\mathrm{enc}}), \tag{18}$$

$$\mathrm{SelfAttnDec}(\mathbf{H}_D^{\mathrm{enc}}, \mathbf{H}_I^{\mathrm{enc}}) = \mathrm{MultiHead}(\mathbf{H}_D^{\mathrm{enc}}, \mathbf{H}_I^{\mathrm{enc}}, \mathbf{H}_I^{\mathrm{enc}}). \tag{19}$$

The output of the decoder $\mathbf{h}_t^{\mathrm{dec}}$ is the subtopic representation of the document d_t. This representation models the subtopic coverage of document d_t. The remaining part of the decoder component is similar to the encoder component, including the feed-forward network, ReLU activation function, and layer normalization.

4.4 Modeling Global Document Interactions via Self-Attention

Based on self-attention networks, the static feature component of GDESA can globally measure the interactions among all the candidate documents. The global interaction can make the GDESA model outperform the previous models, especially when assessing former ranking positions. We expand on the example used in Section 1 to explain. Assuming there are three candidate documents d_1, d_2, d_3, with d_1 covering the subtopic q_1, d_2 covering q_2, q_3 and d_3 covering q_1, and the three documents have similar relevance scores to the given query. The subtopics provided to the model are not identical to actual user intents and the model cannot measure the intent coverage precisely. In the view of global interactions, d_1 and d_3 are redundant because they are similar with each other, while d_2 is novel comparing with both d_1 and d_3. As a result, GDESA with global interactions can return a higher ranking score for d_2. Then, d_2 is put on a former ranking position.

Both DESA and GDESA apply a self-attention mechanism to model the global interaction between documents. However, the self-attention itself cannot distinguish whether a document has been selected or not, because all documents are computed simultaneously. Under this circumstance, it is hard for the model to predict the position of each document precisely as some document selection is heavily relied on previous selection. To tackle this problem, we add a sequential interaction component to simulate the selection process. The global interaction is only conducted between candidate documents, so the selection is also relied on their relationship with the previous selected document list. By this means, our GDESA can inherit some advantages of greedy document selection. The whole process can better meet the goal of search result diversification, namely satisfying more user intents at former ranking positions.

5 OPTIMIZATION AND ANALYSIS

In this section, we will introduce the model training and optimization of GDESA. We further analyze the mechanism of self-attention diversification task in theory. Finally, we will compare our framework with previous works and discuss the differences between them.

5.1 Training and Optimization

We first introduce the optimization process of GDESA. As we described in Algorithm 1, for each ranking position, GDESA takes all documents in the sequence $\mathcal{D}$, all subtopics $\mathcal{I}$, and the selected sequence context C as input, and returns the ranking scores of all the unselected candidate documents for greedy document selection. In the training phase, the score of a ranking r is calculated by summing up all the scores of documents in r:

$$s_r = \sum_{i=1}^{|r|} s_i. \tag{20}$$

5.1.1 List-pairwise Sampling. Since datasets for search result diversification task are limited, we use the list-pairwise sampling approach proposed by Jiang et al. [15] to obtain sufficient training samples. We use pairs of training samples (C, d_1, d_1) with common context C. The documents d_1 and d_2 in the pair are concatenated with C to generate the document sequence pair r_1 and r_2 respectively, so that the metric (*e.g.*, α-nDCG) of the positive document ranking sequence $M(r_1)$ is better than the negative ranking sequence $M(r_2)$.

The sampling process is described as follows: we first obtain a group of contexts C with different lengths, then the rest of the candidate documents are traversed, sampling a pair of document (d_1, d_2) where the metrics of $[C, d_1]$ and $[C, d_2]$ are different. Here, some contexts C are obtained from the ideal rankings generated by human labeled user intents annotations, and the others are sampled randomly. When using the list-pairwise samples, the original loss function can be defined as a binary classification log-loss formation:

$$Loss = \sum_{q \in Q} \sum_{s \in S_q} |\Delta M| [y_s \log(P(r_1, r_2)) + (1 - y_s) \log(1 - P(r_1, r_2))] \tag{21}$$

where s is a pair of samples; S_q is all sampled pairs of query q; Q is the set of all the queries; $y_s = 1$ for positive and 0 for negative; and $P(r_1, r_2) = \sigma(s_{r_1} - s_{r_2})$ is the probability of being positive. $\Delta M = M(r_1) - M(r_2)$ represents the weights of this sample, meaning that if the metric gap between the positive and negative rankings is larger, the sample is more important. This process can be seen as a simulation of document selection process. With the given context C, selecting the positive document d_1 rather than d_2 will be more beneficial to the diversity of document sequence since $M(r_1) > M(r_2)$. Training with pairwise sample (d_1, d_2) makes the model prefer to select a better document, which is the same as the ranking process of greedy document selection.

5.1.2 Sequence Mask for Training. In the training phase, both positive and negative samples are the ground-truth rankings, not the candidate document sequence. Hence, the self-attention component is modified with a sequence mask used in the original Transformer decoder structure. Similar to users' behavior, the diverse ranking task is a top-down process, and the evaluation metrics of the document at the position i should not be affected by the document at the position $j (j > i)$. Taking a pair of list-pairwise samples as an example: for the pair $[C, d_1]$ and $[C, d_2]$ mentioned in Section 5.1.1, the document representation $\mathbf{h}_d^{enc}$ of the document d $(d \in C)$ will not be affected by the appended document d_1 or d_2.

The sequence mask can prevent the unexpected self-attention interaction and guarantee that each document only interact with itself and the documents at former positions. The scores of documents at former positions will not be affected by the documents at latter positions. The sequence mask only takes effect in the training phase.

5.1.3 Context-based Pairwise Loss. As we described in Equation (20), the scores of a ranking r is the sum of all document ranking scores in the sequence. For the sampling pair $r_1 = [C, d_1]$ and $r_2 = [C, d_2]$, we have got $s_{r_1} = \sum_{i \in C_1} s_i + s_{d_1}$ and $s_{r_2} = \sum_{i \in C_2} s_i + s_{d_2}$. Here, $C_1 = C_2 = C$. Equation (7) shows that the ranking score of a candidate document d is determined by those following components: the relevance features $\mathbf{x}_{d,q}$ and subtopic scores $\mathcal{S}_{d,\mathcal{I}}$, the self-attention based document representations $[\mathbf{h}_d^{\text{enc}}; \mathbf{h}_d^{\text{dec}}]$, and the RNN-based hidden state $\mathbf{h}_d^{\text{DS}}$. $\mathbf{x}_{d,q}$ and $\mathcal{S}_{d,\mathcal{I}}$ are the mono-variate features of d which will not be influenced by other documents, so we get $\mathbf{x}_{C_1,q} = \mathbf{x}_{C_2,q}$ and $\mathcal{S}_{C_1,\mathcal{I}} = \mathcal{S}_{C_2,\mathcal{I}}$. Due to the sequential property of recurrent neural networks, we can also get $\mathbf{H}_{C_1}^{\text{DS}} = \mathbf{H}_{C_2}^{\text{DS}}$. In Section 5.1.2, we mention that a sequence mask is deployed into the self-attention encoder, and it will strictly guarantee that $\mathbf{H}_{C_1}^{\text{enc}} = \mathbf{H}_{C_2}^{\text{enc}}$. Since r_1 and r_2 are under the same query, we have $\mathcal{I}_1 = \mathcal{I}_2$ and $\mathbf{H}_{\mathcal{I}_1}^{\text{enc}} = \mathbf{H}_{\mathcal{I}_2}^{\text{enc}}$, so it can also guarantee that $\mathbf{H}_{C_1}^{\text{dec}} = \mathbf{H}_{C_2}^{\text{dec}}$.

In conclusion, for the documents in common context C, it can strictly guarantee that $\sum_{i \in C_1} s_i = \sum_{i \in C_2} s_i$. So, we can get:

$$s_{r_1} - s_{r_2} = s_{d_1} - s_{d_2}, \tag{22}$$

$$P(r_1, r_2) = P(d_1, d_2). \tag{23}$$

Denoting the binary classification log-loss function as LogLoss, Equation (21) can be simplified as:

$$Loss = \sum_{q \in Q} \sum_{[C,(d_1,d_2)] \in S_q} |\Delta M| \text{LogLoss}(P(d_1, d_2)). \tag{24}$$

This is the definition of the context-based pairwise function. For search result diversification task, the scores of d_1 and d_2 depend on the context C. However, since the metrics of the context documents should not be affected by the latter documents, the ranking scores of $\sum_{i \in C} s_i$ will not affect the loss function. This means that the context-based pairwise loss function can be defined in the form of a pairwise loss function for the document pair (d_1, d_2). It must be addressed that this loss function is actually a kind of listwise loss function since s_{d_1} and s_{d_2} depend on the whole ranking lists of the documents. The target of the model optimization is to maximize the distance between the positive document d_1 and negative document d_2. When the model is trained, the goal of the optimization is to improve the model's ability to indicate whether a single document in the candidate sequence is novel and covers more subtopics than the other candidate documents.

5.2 Theoretical Analysis

In this section, we analyze the effect of self-attention in the encoder-decoder structure of GDESA. Here, we describe in detail why self-attention is suitable in the diverse ranking task. For simplicity, we will first focus on a single-layer self-attention function in the encoder component and ignore those additional strategies, *e.g.*, positional embedding, multi-head attention, or layer normalization.

The self-attention interaction of the document sequence D is calculated in parallel as an entire matrix, and the attention score can be written as the following equation focusing on the t-th document d_t represented as q_t, discarding the scalar factor $\sqrt{d}$:

$$\text{Score}_{\text{Attn}}(\mathbf{Q}_t, \mathbf{K}) = \text{softmax}(\mathbf{Q}_t \mathbf{K}^\top). \tag{25}$$

For self-attention, it can be approximated that $\mathbf{Q} = \mathbf{K} = \mathbf{V} = \mathbf{D}$, and $\mathbf{q}_t \approx \mathbf{d}_t$. The $\mathbf{q}_t \mathbf{K}^\top$ in Equation (25) can be seen as the dot product scores between the t-th document and each document in the sequence including itself. Using the softmax function, these scores are converted into weights. Since the dot product of two documents can represent the similarity score between the two documents, the weights model the similarity distribution between d_t and each document in the sequence. The self-attention output of d_t is defined as follows:

$$\mathbf{h}_t^{\text{enc}} = \text{softmax}([s_1, \ldots, s_n])^\top \mathbf{V} = [w_1, \ldots, w_n]\mathbf{V} = \mathbf{W}_t^\top \mathbf{V}. \tag{26}$$

Here, n is the length of document sequence, s_i is the dot product between document d_t and d_i, and w_i is the similarity weight converted from s_i. Section 5.1.1 shows the details of list-pairwise sampling for training. With shared selected context document sequence C, positive and negative document pair d_{pos}, d_{neg}, the positive and negative samples can be written as $[C, d_{\text{p}}]$ and $[C, d_{\text{n}}]$. Given the softmax function property, $\sum_{i=1}^{n} w_i = 1$, for the weights distribution of document d_t, the equation can be written as:

$$\sum_{i \in C} w_i + w_t = 1. \tag{27}$$

In the view of MMR, compared with the context C, the positive document d_{pos} should be a novel document, which means that d_{pos} should be dissimilar with the documents in the context C. The dot product scores of d_{pos} with other documents $d_i (i \in C)$ should be significantly smaller than the scores of d_{pos} with itself, indicating $s_{\text{pos}} \gg \sum_{i \in C} s_i (i \in C)$. After the softmax function, we get $w_{\text{pos}} \gg \sum_{i \in C} w_i (i \in C)$. For the negative document d_{neg}, since it is a redundant document, the dot product scores with the context documents will be close to the score with itself, and $w_{\text{neg}} \gg \sum_{i \in C} w_i (i \in C)$ is no longer valid. As a result, a positive document will gain an attention distribution concentrated to the document itself, while a negative document will gain an average distribution. This is identical to the spirit of MMR, since a novel document should be dissimilar with the other documents, and its similarity scores with other documents should be much smaller than the score with itself. Figure 3 shows the different attention distributions of novel and redundant documents.

Using context-based pairwise optimization, the attention distribution distance between the positive and negative documents will increase, and the learning-to-rank function of the model will be trained to return a ranking score of d_{pos} higher than d_{neg}. With more self-attention layers, the distribution distance between the positive and negative samples will expand further, and the learning-to-rank function will be more effective to judge the novelty of a candidate document.

The analysis above is based on the self-attention encoder. Similarly, the document representation of self-attention decoder can be defined as the following equations:

$$\mathbf{h}_t^{\text{dec}} = \text{Attn}(\mathbf{h}_t^{\text{enc}}, \mathbf{H}_I^{\text{enc}}, \mathbf{H}_I^{\text{enc}}) = [w_1^{\text{dec}}, \ldots, w_n^{\text{dec}}]\mathbf{H}_I^{\text{enc}} = (\mathbf{W}_t^{\text{dec}})^\top \mathbf{H}_I^{\text{enc}}. \tag{28}$$

Here, w_i^{dec} denotes the attention weights between document d_t and subtopic q_i. Similar to the encoder attention distribution, the decoder attention distribution of $\mathbf{h}_t^{\text{dec}}$ will focus on the subtopics relevant to d_t, and the irrelevant subtopics will be ignored with lower attention weights. The decoder attention distributions of positive and negative documents will be similar to the encoder attention. For the positive document, the attention distribution will be concentrated on the relevant subtopics, and for the negative document, the distribution will be averaged since none of the subtopics are relevant to the document. The decoder takes the encoded output representations of documents $\mathbf{H}_{\text{D}}^{enc}$ and subtopics $\mathbf{H}_{\text{I}}^{enc}$ as input. A redundant document will also be affected by its encoder

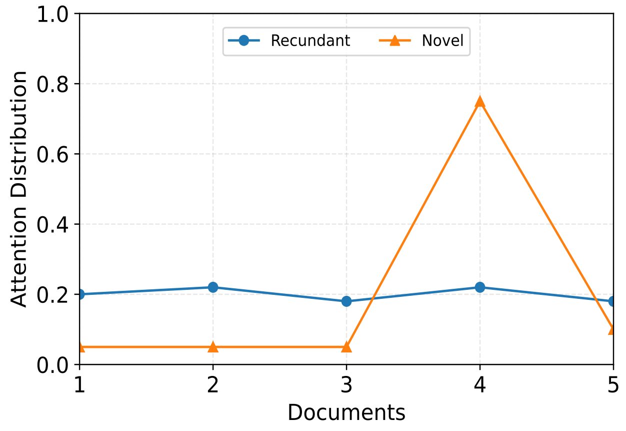

Fig. 3. Different attention distribution of novel and redundant documents

attention distribution, as its decoder attention distribution will be averaged more than the decoder attention distribution for its original representation. This effect is also valid for subtopics. For a subtopic q_i with redundant encoder attention, since $\mathbf{h}_{q_i}^{\text{enc}} = (\mathbf{W}_{q_i}^{enc})^{\top}\mathbf{I}$, its corresponding decoder attention weights w_i^{dec} will also be affected and weakened through the average distribution of $\mathbf{W}_{q_i}^{\text{enc}}$.

Taking $\mathbf{h}_t^{\text{dec}}$ as input, the learning-to-rank function will be able to model the subtopic coverage of $\mathbf{d}_t$ together with the relevance scores of subtopics.

5.3 Comparison with Previous Work

GDESA is inspired by several previous works: sequential-interaction based search result diversification approach, *e.g.*, DSSA, and global-interaction approach, *e.g.*, SetRank and our previous work of DESA. In this section, we will compare GDESA with different types of previous works and analyze their differences in detail.

5.3.1 GDESA v.s. Sequential-based Supervised Learning Approaches. The typical state-of-the-art supervised learning approaches include the following ones: PAMM-NTN [35] for modeling the implicit diversity features, DSSA [15] for modeling explicit diversity features, and DVGAN [18] for integrating both implicit and explicit features. Ideally, the model with explicit features should always perform better than the model with implicit features. But in practise, the explicit features are usually extracted from documents and mined subtopics, which may not be identical to the actual user intent coverage. As a result, it will be optimal to combine both explicit and implicit features. DSSA, DVGAN, and GDESA are sharing the unsupervised document and subtopic embeddings. While DSSA is an explicit model and it does not measure the implicit diversification features among documents. Both GDESA and DVGAN combine the implicit and explicit features together to achieve better diversification results. For DVGAN, two instances of diversification models are used as the generator and discriminator of GAN. Usually the generator is an explicit model and the discriminator is an implicit model. And for GDESA, the implicit and explicit features come from the self-attention and cross-attention encoders based on document and subtopic representations.

Besides, all those supervised approaches mentioned above purely depend on sequential interactions and greedy selection algorithms, which may lead to local optimal ranking results. In a global view, these results may be suboptimal. While the GDESA framework ensembles both sequential and global interactions, which can minimize the gap between local optimal and global optimal rankings.

5.3.2 GDESA v.s. Global-based Supervised Learning Approaches. Typical multivariate ranking approaches include DLCM [2], DIN [21], and SetRank [20]. DLCM is based on RNNs and attention mechanism, and the rest works are based on self-attention networks. All those approaches mentioned are designed for ad-hoc ranking tasks, and the corresponding diversification extensions include DALETOR [39] and DESA [22]. All these global-based approaches, for both ad-hoc ranking and diversification, are not relied on sequential document selection. They do not distinguish whether a document is selected or not. Since the self-attention network is permutation equivalent [20], both DESA and DALETOR may overlook the ranking positions in diversification tasks. Although DALETOR can implicitly model the sequential interactions via optimizing the diversity metric, it is still beneficial to model the sequential interactions more explicitly with an independent component. GDESA is based on DESA, and it includes a document selection component for modeling the sequential interaction between the selected document sequence and different candidate documents.

5.3.3 GDESA v.s. MDP-based Reinforced Learning Approaches. Typical reinforced learning (RL) based approaches include MDP-DIV [36] and M2DIV [12], and MDP-DIV is further extended with PPG [38] (Pairwise Policy Gradient) in order to reduce the variance of model training and measure the relative goodness of the documents. We discuss those works in the following sections.

MDP-DIV is a sequential-based approach, and the candidate documents are selected via Markov Decision Process (MDP). As a reinforced learning approach, MDP-DIV formulates the diverse ranking task as a process of sequential selection and decision. In the view of reinforced learning, the state $\mathcal{S}$ represents the current condition of the selected sequence and candidate documents, and the action $\mathcal{A}$ denotes the operation of selecting a candidate document and appending it into the selected sequence. For each action, the reward r is denoted as the increment of evaluation metric (*e.g.*, α-nDCG) corresponding to the action of document selection. With the policy function π, the model tries to select the document with the best rewards in order to achieve an optimal ranking. The advantage of MDP-DIV is that it can measure both the immediate rewards and the long-term returns by traversing all the candidate documents in model training. Besides, it can make use of the additional information utility for each document selection, and its light-weighted policy function makes it easy to re-rank a large scale of documents in the initial ranking sequence.

However, compared with GDESA, MDP-DIV has got the following disadvantages:

First of all, MDP-DIV is still based on greedy selection, and it does not measure the global document interactions. As a result, MDP-DIV will also lead to local optimal rankings instead of global optimal rankings. Second, as an implicit diversification approach, MDP-DIV cannot take advantage of external subtopic information. Moreover, the policy function of MDP-DIV is completely based on unsupervised document embeddings, *e.g.*, doc2vec, and the relevance features are omitted. Xia et al. [36] claimed that MDP-DIV has got the advantage of end-to-end diverse ranking without any handcrafting features. But in fact, Jiang et al. [15] have also claimed that the quality of unsupervised embeddings is insufficient to model the relevance between queries (including subtopics) and documents. This may be because DSSA is based on the poor Lemur initial rankings. For poor initial ranking lists, the traditional IR features, *e.g.*, BM25 are still necessary to judge if a document is relevant.

Search result diversification aims to satisfy user intents at former ranking positions. If a document satisfies any user intents behind the query, it must be relevant to this query. A "diversified" but irrelevant ranking list will not satisfy user intents. With a rich initial ranking list, MDP-DIV may perform well since it will be easier to select a relevant document covering intents. But if the ranking list is poor, MDP-DIV may fail to capture the correct document with intent covered, leading to bad performance.

Besides, MDP-DIV is trained with the strategy of policy gradient, and the policy gradient strategy will directly utilize the absolute performance scores instead of relative goodness. This may increase the variance of gradient estimation, and the model may fail to estimate if a document is better than another one. The successor work of PPG focus on resolving the issue of relative goodness by introducing pairwise policy gradient in the model training of MDP-DIV, which is similar to the pairwise loss function used in supervised learning. While the other issues mentioned above are remained.

5.3.4 GDESA v.s. M2DIV. Another successor work of MDP-DIV is M2DIV, here "M2" denotes MDP and MCTS (Monte Carlo Tree Search). Same as GDESA, M2DIV also focuses on minimizing the gap between local and global optimal rankings. In order to achieve this goal, M2DIV adapts MCTS instead of greedy selection to select the next candidate documents in the sequential selection process. Comparing with greedy selection, MCTS can explore the possible rankings in a larger scale of ranking spaces. As a result, the rankings searched by MCTS will get closer to the global optimal rankings. While the critical issue of M2DIV is that MCTS is extremely time-consuming. For online ranking tasks, MCTS may not be feasible since the inference latency is too high. In order to address this issue, M2DIV also includes a raw policy without MCTS for online ranking tasks. Unfortunately, the time consumption of model training is unavoidable. In the condition of commercial search engines, there will be a large amount of training data, and the exploiting time cost of model training will be unaffordable and unacceptable.

Moreover, MCTS is a sequential process, and it is difficult and ineffective to accelerate the model training with typical parallel computing hardware. In the training algorithm of MCTS, the framework selects a node recursively until reaching a leaf node. Then, the node will either be expanded through simulation with value function or evaluated with the ground-truth. Due to the sequential dependency, most of these processes above can only be computed on a single CPU core, and only the simulation part can theoretically be accelerated by parallel computation. But in practice, the implementation of parallel simulation is difficult, and the rest parts of the algorithm cannot be accelerated. Besides, the policy function of M2DIV is only a single layer LSTM, which may not take full advantage of GPU computing. For typical computing machines, the CPU should first copy data from system memory into video memory before GPU computation, and it should also copy the data back into system memory when the GPU has finished computing. Copying data between system memory and video memory should pass the data through the PCI-Express channels, which requires extra time latency. Since the policy function in M2DIV is not large enough, the time saved by GPU computing can barely cover the increasing latency of data pass-through. As a result, using GPU in M2DIV model training cannot significantly increase the training speed.

Compared with M2DIV, GDESA is much more effective and efficient in both online ranking and model training. The core operator in GDESA is Multi-Head Self-Attention (MHSA), which is widely supported by multiple deep learning frameworks, such as PyTorch or LightSeq, leading to easy implementation of the model. The model training of GDESA can be easily accelerated by GPUs or other parallel computing hardwares. For model training, a GPU-accelerated GDESA can be more than 100× faster than M2DIV. It should also be addressed that the training process of all MDP-based models also requires a larger scale of initial ranking sequences as input. As a result, the model training of GDESA is faster than MDP-DIV and PPG.

5.4 Analysis of Time Complexity

In this section, we analyze the time complexity of GDESA. The framework of GDESA can be divided into the static and dynamic components, and only the dynamic components is computed repeatedly in the document selection process. The static component includes the encoder and decoder based on self-attention networks. Ignoring the dimension expansion of document and subtopic embeddings, the time complexity is:

$$\Theta_{\text{stat}} = L_{\text{enc}}|\mathcal{D}|^2 E_d + L_{\text{dec}}|\mathcal{D}||\mathcal{I}|E_d + |\mathcal{I}|R + |\mathcal{I}|E_q. \tag{29}$$

Here, R is the number of relevance features, E_d and E_q are the hidden sizes of document and subtopic embeddings. L_{enc} and L_{dec} denote the number of layers in encoder and decoder respectively.

For the dynamic component, with vanilla RNN cells for document selection, the complexity of each single ranking position is:

$$\Theta_{\text{dyn}} = U(U + E_d) + |\boldsymbol{v}|, \tag{30}$$

$$|\boldsymbol{v}| = 2E_d + R + U + |\mathcal{I}|. \tag{31}$$

Here U is the hidden size of the cell, and $|\boldsymbol{v}|$ is the dimension of the vector $|\boldsymbol{v}|$ for overall scoring of Equation 12. The overall time complexity can be described as:

$$\Theta = \Theta_{\text{stat}} + |\mathcal{R}|\Theta_{\text{dyn}}. \tag{32}$$

In online ranking tasks, as those static features are unaware of the document selection process, the static components will compute only once. Since the search result diversification task is a re-ranking task, the ranking list $|\mathcal{D}|$ will not be too long. In real ranking tasks, $|\mathcal{R}|$ can be smaller than $|\mathcal{D}|$. For example, the model may take the initial ranking list $\mathcal{D}$ with $|\mathcal{D}| = 50$, while the length of diversified ranking list $\mathcal{R}$ can be as small as $|\mathcal{R}| = 20$. Search result diversification aims to satisfy more user intents at former ranking positions. For the TREC official evaluation metrics, at most top 20 documents in the re-ranking lists are taken into consideration. In the real applications, users prefer to pay more attention to the results on the top ranking positions, while ignoring the latter results. On a 40 core 2.2GHz CPU server, the GDESA ranking takes about 44.7ms per query, and on the same server with a Titan V GPU, the latency of a single query is about 24.7ms. Details about the ranking latency will be described in Section 7.5.

6 EXPERIMENTAL SETTINGS

6.1 Data Collections and Evaluation Metrics

6.1.1 Datasets. In the experiments we used the same dataset as previous diversification models (*e.g.*, HxQuAD, PAMM-NTN, and DSSA), which includes the Web Track dataset from TREC 2009 to 2012. There are in total 200 queries and 198 queries are used since query #95 and #100 have got no diversity judgements to use. Each of them includes 3 to 8 annotated subtopics, and the relevance rating is marked as relevant or irrelevant at subtopic level. We conduct all experiments on the ClueWeb09 dataset [7].

The subtopics used by the model come from the Google query suggestions provided by Hu et al. [13], and we only use the first level of the subtopics with no hierarchical subtopics. The maximum subtopic number of the queries is 10, and the average subtopic number is about 9.48. As those previous works did [13], we regard all the subtopics as having uniform weights. For the implicit approaches, the subtopics are ignored.

For a fair comparison, we used the same document relevance features and embeddings as with DSSA, which have been released by Jiang et al. [15] in the repository on GitHub[3]. The training data includes 18 relevance features for each query and subquery produced by traditional IR models, *e.g.*, BM25 and TF-IDF, and the document embeddings are generated by doc2vec with a window size of 5. In future work, we plan to incorporate several deep-learning based technologies for feature extraction and document representation, *e.g.*, K-NRM [37] or BERT [10].

All of our training and evaluation data samples are exactly the same as DSSA. The list-pairwise training samples are produced by the top-20 documents of Lemur initial rankings, and the top-50 initial rankings are used for evaluation. And the pseudo documents for subtopic embeddings are produced by the top-20 (Z) documents. Codes and ranking files can be accessed at: http://github.com/qratosone/GDESA.

[3]https://github.com/jzbjyb/dssa

6.1.2 Evaluation Metrics. The official diversity evaluation metrics of Web Track include ERR-IA [6], α-nDCG [8], and NRBP [4], which are used in our experiments. Besides the metrics above, we also include the metrics of Precision-IA [1] (denoted as Pre-IA), D#-nDCG [26] and Subtopic Recall [42] (denoted as S-rec). Inheriting the spirit of the previous works [15, 18, 34, 35], all the metrics were computed on the top 20 results in the document ranking lists. Two-tailed paired t-test are used to conduct significance testing with p-value<0.05. In the significance testing, GDESA is compared with DSSA as the SOTA explicit supervised model of sequential interactions.

6.2 Model Settings

On our GPU machine, the training phase of GDESA with the training samples of 160 queries can be finished in three hours. We tuned the layer number L of the self-attention network in order to avoid overfitting, with $L = L_{\text{enc}} + L_{\text{dec}}$, where L_{enc} is the number of layer for the encoder component and L_{dec} is the number of layers for the decoder. The layer number of LSTM is fixed into 1 and we only tune different L in cross validation. The batch size is 256 and the learning rate is 0.008. Notice that the hyper-parameter settings of the encoder and decoder in GDESA are not identical to the settings in DESA. Details of those settings will be described in Section 7.3.

We compare our settings with the non-diversified baseline and those of previous implicit/explicit supervised models. The detailed settings of GDESA are described next. We use five-fold cross validation to tune the parameters in all experiments with the widely used metrics α-nDCG@20. The settings of the baseline models are described as follows:

Lemur. We use the search results produced by the language model and retrieved by the Lemur service as the non-diversified baseline.[4] These results are released by Hu et al. [13] and can be found on the website.[5] All the diversification approaches in our experiments use the search results of Lemur as initial ranking sequences.

xQuAD [27], PM2 [9], HxQuAD, and HPM2 [13]. These are the unsupervised explicit baseline approaches for comparison. All the unsupervised methods use the parameter λ to combine the relevance and diversity linearly. HxQuAD and HPM2 require an extra parameter α to control the weights of the hierarchical subtopic layers. The parameters are tuned with cross validation, and ListMLE [41] is used to learn a prior relevance function without diversification.

R-LTR [45], **PAMM** [34], and **PAMM-NTN** [35]. Inspired by previous work [15], we use the metric of α-nDCG@20 to tune the parameters. The neural tensor network (NTN) is combined with both R-LTR and PAMM, which are denoted as R-LTR-NTN and PAMM-NTN, respectively. The number of tensor slices for NTN is tuned from 1 to 10, and the number of positive ranking τ^+ and negative ranking τ^- are tuned per query for the PAMM.

DSSA [15]. DSSA is the previous state-of-the-art explicit supervised diversification approach. It models diversity by subtopic matching scores with attention-based weights. We train the DSSA model with the released code and data, and use the following optimized settings described in the work of DSSA: we use LSTM cells and max-pooling on subtopic attention; the hidden size is 50; the dimension of the doc2vec embedding is 100; and the random permutation counts 10 for the list-pairwise samples. In the released data, the author only provides the document and subtopic embeddings of doc2vec, not the embedding of LDA reported in the work. Since the quality of document embedding is not the focus of our work, for a fair comparison, all the experiments of NTN, DSSA, DVGAN, DESA and GDESA are based on the same doc2vec embeddings. We denote this result as as DSSA (doc2vec).

DVGAN [18]. DVGAN integrates an explicit model and an implicit model with a GAN framework. We use the best setting of DVGAN-doc reported in the paper, namely using DSSA as generator and R-LTR as discriminator and sampling about 10 samples of 10 negative rankings of 20-document-length for each query.

[4]Lemur service: http://boston.lti.cs.cmu.edu/Services/clueweb09_batch/
[5]http://playbigdata.ruc.edu.cn/dou/hdiv/

DESA [22]. DESA is a global-based ensemble approach with both implicit and explicit features. We use use the same configuration in the paper: $L_{\text{enc}} = 2$, $L_{\text{dec}} = 1$ with head num $H = 8$. With the doc2vec embeddings, the projected embedding length is 256 and feed-forward length is 400.

MDP-DIV [36]. MDP-DIV is an implicit diversification approach based on reinforcement learning and Markov Decision Process (MDP). We directly use the released code and configuration files and slightly adjust some of the hyper-parameters.[6] The learning rate is 1e-4, the hidden size is 5, and γ is 1. The lengths of training and inference permutation are expanded to 20. Since the evaluation metrics reported in [36] is based on another initial rankings which are different from the Lemur initial rankings used in our experiments, we report the results based on their initial rankings and our initial rankings, which are denoted as "MDP-DIV (original)" and "MDP-DIV (Lemur)", respectively.[7]

For "MDP-DIV (Lemur)", since it requires a large amount of documents to train the policy function, we use the documents with intent annotations in the top-500 initial ranking list of Lemur. These documents can be obtained by the data preparation tool "prep.py" in Jiang's GitHub Repository, and more details of the tool can be found in the README file of the Repository. The document and query embeddings for "MDP-DIV (Lemur)" are the same as used in DSSA and GDESA. The evaluation metrics used in the original experiments of MDP-DIV is ERR-IA@10 and α-nDCG@10. However, in our experiments, we use ERR-IA@20 and α-nDCG@20.

PPG-DIV [38]. PPG-DIV is an implicit approach based on reinforcement learning and integrates the strategy of pairwise policy gradient (PPG). As the author does not release the data corresponding to their experiments, we only reproduce the experiments on our Lemur initial rankings. Similar to MDP-DIV, we denote these results as "PPG-DIV (Lemur)". Following the discussion parts of the original paper [38], we train the model for 200 epochs. As there are not detailed settings provided in the paper, we directly use the settings released in their code.

DALETOR [39]. DALETOR is a global-based implicit diversification approach relying on optimizing the diversity metric. As the source code is not released, we implement DALETOR by ourselves for the experiments. Following the paper [39], we optimize the model with the loss of α-nDCG. Notice that the results reported in [39] are based on the initial rankings which are the same as "MDP-DIV (original)". We reproduce the experiments of DALETOR only on Lemur initial rankings, the results are denoted as "DALETOR (Lemur)".

7 EXPERIMENTAL RESULTS

7.1 Overall Results

Table 3 shows the overall results of all the models. GDESA outperforms all those implicit and explicit baselines based on sequential interaction features. The performance improvement is statistically significant on all metrics except for Pre-IA and D#-nDCG. These experimental results clearly demonstrate the advantage of GDESA. Comparing the SOTA sequential-based approach, the improvement of GDESA over DSSA on α-nDCG is about 3%. As an explicit model, DSSA uses the RNN and attention mechanism to select the best document satisfying the subtopics needed by the selected sequence, but it does not model the document novelty implicitly. GDESA outperforms DSSA by leveraging both document novelty and subtopic coverage simultaneously. Besides, DSSA purely depends on greedy document selection, which may select the local optimal document at each step leading to a global suboptimal ranking. Conversely, the self-attention networks in GDESA can learn the global document interactions, which can significantly minimize the gap between local and global optimal rankings.

GDESA can also slightly outperforms DVGAN and DESA without document selection. These two approaches can leverage both implicit and explicit features for diversification. While DVGAN purely focuses on sequential interactions and DESA only models global interactions. These results indicate that for the diverse ranking task both sequential interactions and global interactions are of great importance. It is beneficial to leverage both

[6]https://github.com/sweetalyssum/RL4SRD

[7]It should be addressed that the results of "MDP-DIV (original)" are not comparable with those of other baselines.

Table 3. Performance of all approaches. The best results are in bold. ★ indicates that the GDESA model significantly outperforms all implicit and explicit baselines ($p < 0.05$ in two-tailed paired t-test), except DVGAN and DESA. Notice that "MDP-DIV (original)" is not comparable with other baselines.

Methods	ERR-IA	α-nDCG	NRBP	Pre-IA	S-rec	D#-nDCG
Lemur	.271	.369	.232	.153	.621	.424
xQuAD	.317	.413	.284	.161	.622	.437
PM2	.306	.411	.267	.169	.643	.450
HxQuAD	.326	.421	.294	.158	.629	.441
HPM2	.317	.420	.279	.172	.645	.454
R-LTR	.303	.403	.267	.164	.631	.441
PAMM	.309	.411	.271	.168	.643	.450
R-LTR-NTN	.312	.415	.272	.166	.644	.451
PAMM-NTN	.311	.417	.272	.170	.648	.457
DSSA (doc2vec)	.350	.452	.318	.184	.645	.471
MDP-DIV (Lemur)	.307	.402	.274	.156	.606	.430
MDP-DIV (Original)	.375	.503	.333	.234	.706	.547
PPG-DIV (Lemur)	.229	.328	.186	.119	.598	.393
DALETOR (Lemur)	.305	.396	.270	.149	.606	.420
DVGAN	.367	.465	.334	.175	.660	.472
DESA	.363	.464	.332	.184	.653	.475
GDESA	**.369**★	**.469**★	**.337**★	.185	**.662**★	.480

the two kinds of interactions: the global interactions among all the documents, and the sequential interactions between each candidate document and the selected sequence.

7.2 Influence of Initial Rankings

Search result diversification is often conducted as a re-ranking stage based on an initial ranking list. Recall that diversification tasks are designed to suit different user intents at former ranking positions; thus, a diverse but irrelevant result list cannot satisfy users' actual information needs. This is the principle of the evaluation metrics. Using α-nDCG as an example: for document d_i at the ranking position i, if d_i satisfied an intent that has already been satisfied, the value of d_i will be penalized since d_i satisfies a redundant intent. However, if d_i satisfies no intent, the metric of d_i will be 0 as d_i is irrelevant to the query. In other words, for a given query, a relevant document can cover at least one intent, whereas an irrelevant one cannot cover any intent.

As shown in Table 3, we can confirm that:

(1) GDESA can slightly outperform those advanced baselines, *e.g.*, DESA or DVGAN, yet the improvements are not significant.
(2) Based on Lemur rankings, the approaches of MDP-DIV, PPG-DIV, and DALETOR perform badly. The performance of "PPG-DIV (Lemur)" is even worse than "MDP-DIV (Lemur)".

We further check the quality of initial rankings used in our experiments. The rankings used by MDP-DIV, PPG-DIV, and DALETOR are denoted as *original rankings*. We found that in Lemur rankings, there are 16 empty queries among all the given queries, implying that there are no positive documents in the top-50 ranking lists corresponding to the queries. Furthermore, only 8.34 documents on average have positive annotations for the top-50 rankings per query, whereas 22.64 documents have positive annotations for the top-500 rankings. As a comparison, in original rankings, there are an average of 67.06 documents with positive intent annotations per query. Based on these observations, we conduct the following analysis of the influence of initial rankings for diversification approaches.

7.2.1 Ceiling Effect of Initial Rankings. Due to the properties of re-ranking tasks, the performance of diversification models is limited by the initial ranking lists. For diversification metrics, such as α-nDCG, the TREC official evaluation tool normalizes the result with the global best ideal ranking lists, which are composed of all positively annotated documents. However, in practice, the initial ranking list can only cover a subset of positively annotated documents. Since the search result diversification task is a re-ranking task, a diversification model cannot generate a document that is not included in the initial ranking list. As a result, the intent coverage of initial ranking lists has a significant impact on the performance of diversification approaches. For example, as aforementioned, Lemur rankings have 16 dummy queries. Any of those diversification approaches will result in the evaluation metrics for those queries being set to 0, as there are no documents satisfying any of the intents. With such poor initial ranking lists, it is difficult to produce significant improvement when compared to strong baselines such as DVGAN and the original DESA. To our knowledge, even some of the recently proposed approaches [23, 31] based on BERT [10] cannot outperform those strong baselines significantly. We speculate that when a group of better initial ranking lists are given, the model's performance will be further improved. We forgo extensive analysis because it is beyond the scope of this work.

7.2.2 Discussion on Relevance and Diversity. This section discusses the results of MDP-DIV, PPG-DIV, and DALETOR respectively. To make a fair comparison, we use the released code to reproduce the experiments of MDP-DIV and the performance of MDP-DIV (original) is close to that reported by the authors. However, our experimental results indicate that the performances based on the Lemur initial rankings are very bad. The performance of PPG-DIV (Lemur) is even worse than MDP-DIV (Lemur).

The main difference falls into the relevance components. As Xia et al. described in the paper [36], the policy function of MDP-DIV is completely based on the document and query embeddings, and it does not depend on any diversity or relevance features. The policy function of PPG-DIV is similar to MDP-DIV. In DALETOR, the relevance is implicitly measured by using document and query embeddings with the latent cross algorithm to generate the document representations. The common part of those three approaches is that all those three approaches do not include an independent relevance component to measure the relevance of the documents. Those approaches measure the relevance implicitly with the embeddings of queries and documents, omitting the relevance features.

Most of those existing implicit diversification approaches focus on modeling the novelty (or dissimilarity) of documents, while some recent work [31] has already demonstrated that the diversification approaches should directly model the novelty of intent coverage instead of the novelty of documents. For diversification approaches, the relevance components can ensure a document is covering actual intents (relevant), while the relevance signals based on embeddings are not strong enough to judge the relevance of documents [15]. As we discussed above, on average there are about 7 times more positive documents per query in "original rankings" compared with Lemur top-50 rankings. For high-quality initial ranking lists covering rich annotated documents, all those three approaches can outperform PAMM-NTN [35] significantly. However, for Lemur ranking lists there are not enough annotated documents, and those embedding-based approaches without relevance features may fail to distinguish the relevant documents. As a result, those three approaches fail to satisfy user intents when a poor initial ranking list is given.

For PPG-DIV, we also find that the evaluation metrics of the training set can hardly raise when the training process goes on. These results indicate that the pairwise strategy in PPG-DIV is more sensitive to the quality of initial ranking lists. Compared with pairwise learning-to-rank, PPG-DIV is generating the training pairs "online" during model training. In the view of supervised learning, the pairwise policy gradient strategy can be seen as a "learning-to-sample" process. Theoretically, when there are enough annotated documents in initial ranking lists, the "learning-to-sample" pairwise policy gradient strategy can outperform handcrafted pairwise sampling. However, when a poor ranking list is given, it will be difficult to automatically sample high-quality document pairs. As a result, PPG-DIV may suffer more negative effects when a poor initial ranking list is given.

Different from those three approaches, the framework of PAMM-NTN includes a feature-based relevance matching component. This may be the reason why PAMM-NTN can outperform those three approaches on Lemur initial ranking lists. In order to achieve fair performance with poor ranking lists given, we assume that the ranking models of MDP-DIV, PPG-DIV, and DALETOR should be re-designed and enhanced with relevance features. Further exploration is omitted because modifying the model structure of those approaches is irrelevant to our work. As GDESA depends on the relevance features of queries and subtopics, it can achieve significantly better performances on low-quality initial ranking lists, such as Lemur.

7.3 Influence of Different Model Settings

We conducted several experiments to investigate the influence of different settings to the performance of GDESA. As we described above, since GDESA mainly depends on the effect of self-attention, we set the layer number of LSTM cells as 1 and the dimension of hidden states is set as 50. We focus on the self-attention component. The baseline settings of the self-attention component include: the initial document/subtopic embedding is 100 dimensions, and it is projected into 160 dimensions as the input of the self-attention network, d_{FF} is set as 400 in the feed-forward network, and the head number H is 8 for the multi-head attention operation.

We test the effect of different numbers of encoder and decoder layers. The encoder layer number L_{enc} is tuned from 2 to 4 with the decoder layer number L_{dec} ranging from 0 to 2. Experimental results of different settings are shown in Table 4. As can be observed, changing the settings of the self-attention component slightly influences GDESA's performance. In our experiments, we found that the total number of self-attention layer L should be strictly limited in order to prevent over-fitting and achieve good performance. Besides, the decoder layer plays an important role of leveraging subtopic coverage, but too many decoder layers will also harm the overall performance. Our experimental results show that L_{enc}=2 and L_{dec}=1 yield the best performance with the overall layer number $L = 3$.

Following previous work [15], we also investigate the effect of different RNN cells including vanilla cells, GRU and LSTM. Results are also shown in Table 4. The cells of GRU and LSTM yield slightly better performance than vanilla cells. This potentially stems from the advantage of modeling long-range dependencies for GRU and LSTM cells. We also find that the difference is relatively small. Similar results are reported by DSSA [15].

7.4 Influence of Subtopic Settings

GDESA use the decoder component to generate the decoded document representations indicating the coverage of the subtopics. In order to check the effect of subtopic-related components, we conduct some experiments to evaluate the performance of different subtopic-related model settings. These settings are listed as follows. Notice that "DS" means that the setting includes the document selection component, and "No DS" means that the setting does not include the component.

(1) No Subtopics (DS). Both the decoder component and the subtopic scores are removed. The model works as an implicit model using the document novelty only.
(2) Scores only (No DS). The decoder component is removed and only the subtopic scores are used. All subtopics are enhanced with the encoder-based weights.
(3) Scores only (DS). This setting is the same as "Scores only (No DS)" except that it contains the document selection component.

Table 4. Performance of GDESA with different settings.

Settings	ERR-IA	α-nDCG	NRBP	Pre-IA	S-rec
L_{enc}=2, L_{dec}=0	.362	.460	.332	.181	.654
L_{enc}=3, L_{dec}=0	.365	.466	.333	.182	.660
L_{enc}=4, L_{dec}=0	.361	.461	.329	.180	.655
L_{enc}=2, L_{dec}=2	.363	.465	.329	.185	.665
L_{enc}=1, L_{dec}=2	.365	.466	.332	.184	.658
L_{enc}=2, L_{dec}=1	**.369**	**.469**	**.337**	.185	**.662**
No Subtopics (DS)	.348	.453	.314	.180	.654
Scores only (DS)	.365	.466	.333	.182	.660
Scores only (No DS)	.360	.461	.328	.182	.655
Original Subtopics (DS)	.353	.456	.320	.182	.656
Encoded Subtopics (No DS)	.365	.465	.333	.180	.659
Vanilla	.366	.466	.336	.179	.659
GRU	.367	.468	.336	.182	.660
LSTM	**.369**	**.469**	**.337**	.185	**.662**
Original DESA	.363	.464	.332	.184	.653
GDESA	**.369**	**.469**	**.337**	.185	**.662**

(4) Original subtopics (DS). Both the decoder component and the subtopic scores are used. All subtopics are enhanced with encoder-based weights. The decoder takes the original subtopic representations as input, but not the encoded subtopics.
(5) Encoded weights (No DS). Both the decoder component and the subtopic scores are used. All subtopics are enhanced with encoder-based weights. This setting can be seen as a variation of DESA enhanced by the encoder-based subtopic weights.

For the settings of "No Subtopics" and "Scores only" without decoder components, we use the optimized settings of $H = 8$ and $L = L_{\text{enc}} = 3$. As for the other settings, we use the optimized setting of $L_{\text{enc}} = 2$ and $L_{\text{dec}} = 1$. Experimental results are shown in Table 4. The "No Subtopics" setting is an implicit version of GDESA. Compared with other settings of GDESA, this result demonstrates that the subtopic coverage information plays an important role in GDESA framework. It is important to integrate implicit and explicit features for search result diversification to cover both document novelty and subtopic coverage. While the α-nDCG metric of this setting is close to DSSA (doc2vec) as a explicit baseline, indicating the advantage of GDESA's ensemble framework in leveraging both global interactions and document selection.

Furthermore, these results can also prove that the self-attention encoder can be used to reduce the latent redundancy of the subtopics. As we described in Section 6.1.1, the average subtopic number is 9.48 among all the queries, however, the actual user intent numbers are only 3 to 8, which are smaller than the subtopic numbers in diverse ranking task. These results indicates that the subtopics used in ranking process may still contain redundancy and mislead the diversification model.

We can take the query #1 "obama family tree" in the TREC WebTrack dataset as an example. There are ten subtopics based on Google query suggestions, while there are only three actual user intents in the TREC official subtopic annotations. There are two subtopics q_1 for "obama family tree pictures" and q_2 for "obama family tree photos" in the query suggestions. While both q_1 and q_2 correspond to the same user intent, the redundant subtopics may mislead the model to select a document which covers the different subtopics and increase the actual redundancy.

Following DESA, GDESA take the subtopic embeddings $\mathcal{I}$ as the input to self-attention encoder to obtain the encoded representations. The representations are used as part of the decoder inputs, and we further re-use those representations to generate the encoder-based subtopic weights. These two approaches can both overcome the latent negative effect of subtopic redundancy. Compared with original subtopic embeddings, the encoded subtopic representations can integrate the subtopics' encoder attention distribution into the decoder attention. Similar to the document's attention distribution, the subtopic's attention distribution can also indicate the redundancy of a subtopic. The decoder attention of the redundant subtopic will be affected by the encoder attention. As a result, the decoder attention will be adjusted to reduce the negative effect of the latent subtopic redundancy to the diverse ranking task.

Similarly, the encoder-based weights can also reduce the effect of subtopic redundancy. The S_I in Equation (11) can be seen as the ranking score of subtopics, and a redundant subtopic will be punished by a smaller score and subtopic weight. In original DESA, the subtopic weight $w_j (j \in [1, k])$ is a uniform weight of $1/k$. In our experiments, the α-nDCG metric of "Scores only (No DS)" is close to the metric of original DESA. This result indicates that when there are no decoder in the framework, the encoder-based subtopic weights can take similar effort to reduce the subtopic redundancy.

7.5 Analysis of Inference Time

This section analyzes the inference time of GDESA. We compare GDESA with some previous methods, including the sequential-based DSSA [15], DVGAN [18], MDP-DIV [36], and the global-based DESA [22].

7.5.1 Experimental Settings. For all those models except MDP-DIV, the inference experiments are reproduced on both CPU and GPU environments. As the policy function of MDP-DIV is too small to take the advantage of GPU computing, we only reproduce the experiments of MDP-DIV on CPU. As we described in Section 5.4, we reproduce the experiments on a single server with 40 CPU cores and a single Titan V GPU. All those models except MDP-DIV are implemented by PyTorch, and the GPU will be disabled when measuring the CPU-based inference time. For GDESA, we separately analyze the computational time of static features, and the results are denoted as "GDESA (Static)."

All those models will take the top-50 initial ranking lists as input and re-rank the documents to generate the top-20 diversified ranking lists. We will measure the average inference latency per query. For MDP-DIV, we conduct another experiments taking the top-200 initial ranking lists as input and generate the top-20 ranking lists.

7.5.2 Results and Discussions. Our experimental results are shown in Table 5. These results indicate that GDESA can work efficiently on both CPU and GPU environments. On CPU environments, the inference latency per query is only 46ms, and GPU acceleration can further reduce the latency to 22ms. As a deep-learning based diversification model with sequential selection, GDESA is about 2× faster than DSSA and DVGAN on CPU and 6× faster on GPU.

In Section 3.5, we mentioned that only the dynamic features in GDESA should be computed repeatedly during the document selection process. The static features $\mathrm{Stat}(\mathcal{D}, q, \mathcal{I})$ are unchanged during the document selection process and are computed only once. For DSSA and DVGAN, the entire model has to be computed repeatedly. The dynamic component in GDESA includes only a single RNN cell, while DSSA includes both RNN cells and a bilinear subtopic attention component. DVGAN includes both a DSSA model and an implicit model, *e.g.*, PAMM-NTN or R-LTR. As the dynamic component in GDESA is significantly lighter than DSSA and DVGAN, the computing speed of GDESA for document selection is much faster than DSSA and DVGAN. Besides, the computing of static

Table 5. Results of inference time latency.

	GDESA		GDESA (Static)		DSSA		DVGAN		DESA		MDP-DIV	
Setting	GPU	CPU	GPU	CPU	GPU	CPU	GPU	CPU	GPU	CPU	Top-50	Top-200
Time (ms)	22	46	7	16	145	155	146	154	6	14	8	22

features mainly depends on self-attention networks, which is suitable for parallel computing. For DSSA and DVGAN, as all components required repeatedly computing, the inference latency on GPU is similar to the latency on CPU. For GDESA, using GPU to accelerate the computing of GDESA can reduce almost 50% of the inference time. These results indicate the advantage of GDESA.

Besides, the inference speed of MDP-DIV and DESA is faster than GDESA. DESA is purely based on global interaction, it can be seen as a subset of GDESA with static features only. In Table 5, it can be seen that the computational time of static features in GDESA is comparable to the overall time of DESA. Since the sequential selection process is not used in DESA, it can fully take the advantage of GPU-based parallel computing. MDP-DIV depends on a light-weighted policy function for MDP-based sequential selection. It does not include a large-scale deep neural network, so the overall time complexity is also low for a CPU-based machine. As the absolute inference time of GDESA is acceptable, it is beneficial to use GDESA to achieve better performance.

7.6 Analysis of Model Training Time

In this section, we analyze the training time of GDESA and compare it with M2DIV. We train both models on a single fold with 160 training queries and 40 test queries. For GDESA we train the model with all the list-pairwise training samples for at most 10 epochs. For M2DIV, inspired by the discussion part of the paper [12], we train the model for 100 epochs based on the released code. We only measure the time for training, and the time for evaluation is not included. The overall training time of GDESA is measured on GPU excluding the time of I/O, and the time of M2DIV is measured on both CPU and GPU. As a comparison, we also report the results of DSSA on GPU and MDP-DIV (Lemur) on CPU. Results are listed in Table 6.

These results show that the training of GDESA is much more efficient than M2DIV. The training time of M2DIV is about 80× more than that of GDESA. In real-world application of commercial search engines, the scale of training data are large. Therefore, even for offline training, the booming time cost is unaffordable and unacceptable. We also find that the time cost is reduced little when a GPU is applied for M2DIV model training. These results are consistent with our analysis in Section 5.3.4. The model training process of M2DIV requires repeatedly document selection and Monte-Carlo simulation, and this process is extremely time-consuming. Compared with MDP-DIV, the training time of M2DIV is about 10× more than that of MDP-DIV. Moreover, each selection step depends on the completion of previous steps, leading to the difficulty for parallel computing. When the model is trained on GPU, the RL controller has to repeatedly transfer data between system memory and GPU memory. As the policy function of M2DIV is not large enough to benefit from parallel computing, the saved time of GPU computing may barely cover the increasing latency of passing data through PCI-E channels. These results demonstrate again the advantage of GDESA. Similar to other supervised models such as DSSA, GDESA does not require a CPU-based controller, all the components of the framework can be trained end-to-end on GPU. Besides, the operator of multi-head self-attention is suitable for parallel computing, and it can be easily accelerated via GPU computing. When a large-scale of training data are given, GDESA can be trained and accelerated with multiple GPUs. Compared with DSSA, the training speed of GDESA is slightly slower because the overall network structure of GDESA is larger than DSSA. While the component of self-attention encoder and decoder are suitable

Table 6. Results of model training time. We omit the training time of DESA as it's almost the same as GDESA.

	GDESA		DSSA		M2DIV		MDP-DIV (Lemur)	
Setting	GPU	CPU	GPU	CPU	GPU	CPU	GPU	CPU
Time (hours)	2	-	1.5	-	210	222	-	20

Table 7. Metrics improvement per ranking position. "Total Imp." denotes the total improvement of DESA on all the 200 queries, "Avg Imp." denotes the average improvement per position.

	ERR-IA			α-nDCG		
	@5	@10	@20	@5	@10	@20
DSSA	.328	.344	.351	.400	.428	.452
GDESA	.346	.361	.368	.418	.443	.469
Total Imp.	3.60	3.40	3.40	3.60	2.06	2.37
Avg Imp.	.720	.340	.170	.720	.300	.170

for parallel computing, the parameter increasing of GDESA does not lead to a significantly increase in training time.

7.7 Effect Analysis for Global Interaction

As described in Section 4.4, with the help of global document interaction, our proposed GDESA framework can perform better than the greedy document sequential selection based model at former ranking positions. We analyze the effect of global document interactions in GDESA at different ranking positions. We compare GDESA with DSSA, which is the state-of-the-art sequential selection based model. In this experiment, we use the ERR-IA and α-nDCG metrics computed on top 5, top 10, and top 20 results of a document ranking list to check the effect of GDESA at former ranking positions. Results are shown in Table 7. We calculated the total metric improvement of DESA compared with GDSSA. For simplicity, we use the sum of the metrics for all 198 queries instead of the mean metrics, denoted as Total Imp. We calculate the average metric improvement per position to measure the improvement that GDESA delivered at different ranking position ranges, and this value is denoted as Avg Imp. For example, the ERR-IA@5 of GDESA and DSSA (doc2vec) is 0.346 and 0.328, the total improvement of ERR-IA@5 is calculated as $(0.346 - 0.328) \times 200 \approx 3.60$, and the average improvement of ERR-IA@5 is $3.60/5 \approx 0.720$.

From the results, it can be seen that the average metric improvement values per position for short ranking lists are more significant than those for longer ranking lists. The experimental results are similar to the analysis in Section 4.4, indicating that in the early stage of ranking, the models merely based on document selection may fail to select the globally best candidate document for a short or empty selected sequence. For GDESA, the self-attention networks can measure all candidate documents globally and promote the diversified documents at former positions, satisfying the user intents earlier compared to the models merely based on document selection.

7.8 Effect Analysis for Sequential Property of Self-attention

In this section, we discuss the effect of sequential information in the self-attention component. In GDESA framework, the self-attention encoder for global interaction contains a trainable positional embedding, and the sequence mask mentioned in Section 5.1.2 is not used during the ranking process. In the ranking phase, the document sequence in the self-attention component remain static, and we conduct an experiment with the

Table 8. Influence of sequential settings in self-attention networks.

Methods	ERR-IA	α-nDCG
No-posenc	.358	.458
Static	**.368**	**.469**
Dynamic	.365	.466

Table 9. Evaluation results per query.

Query ID	M(DESA)	M(GDESA)	ΔM
36	.344	.638	.293
174	.314	.593	.279
193	.393	.633	.239
94	.191	.369	.178
138	.469	.642	.173

GDESA framework using a dynamic document sequence in self-attention component as a comparison. Results are shown in Table 8. "no-posenc" denotes the variant of self-attention without positional embeddings, and "static" denotes our full model. "dynamic" denotes the variant in which the selected document d is appended into the selected sequence C, and the input $\mathcal{D}'$ to the self-attention network is $\mathcal{D}' = [C; \mathcal{D}_{\text{rest}}]$. $\mathcal{D}_{\text{rest}}$ is the rest candidate document sequence in which the selected document d is removed from the candidate sequence.

From the result, we can see that the positional embeddings are important for achieving good performance, while using the dynamic document sequence that is changing with the document selection process causes slightly performance loss. Similar to the ordinal embeddings in SetRank [20], the positional embeddings in GDESA framework provide the sequential information of the documents in initial ranking $\mathcal{D}$, not in diversified ranking $\mathcal{R}$. Because the initial ranking is a relevance ranking without diversification, the sequence information for the self-attention network's input sequence represents the documents' relevance rather than the position information for a diversified ranking sequence $\mathcal{R}$. Besides, SetRank [20] has demonstrated that the self-attention network is permutation equivalent. As a result, the self-attention component in our framework is unaware of the position information in the diversified ranking list $\mathcal{R}$. The experimental results reflect the advantage of utilizing an RNN-based component over a self-attention network for document selection.

7.9 Effect Analysis on Document Selection Per Query

In this section, we validate the effectiveness of document selection component in GDESA. We compare GDESA with a variant of DESA that is enhanced by the encoder-based weights. We analyze the performance of the two models per query with the typical evaluation metric α-nDCG@20, results are shown in Table 3. We use M(GDESA) and M(DESA) to denote the evaluation metric of two model.

In all the 198 queries from the dataset, for most of the queries, M(GDESA) is close to M(DESA). There are in total 12 queries which $\Delta M > 0.1$ ($\Delta M = M(\text{GDESA}) - M(\text{DESA})$). Due to space limitation, we select the top-5 queries, and show their evaluation results in Table 9.

The results in Table 9 show that the GDESA-advantage queries have got poor user intent coverage for the candidate documents. Specifically, all of the M(DESA) metrics are lower than 0.5. Focusing on the query #36, Figure 4 shows the α-nDCG variation of GDESA and DESA at each position, where x-axis denotes the ranking position, and y-axis denotes α-nDCG corresponding to each position.

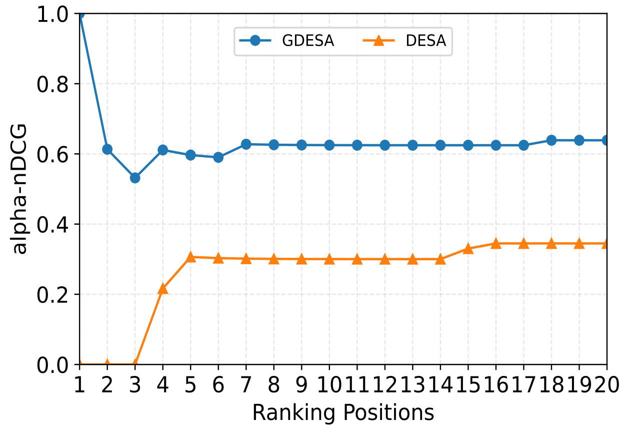

Fig. 4. α-nDCG variation with different positions for GDESA and DESA.

From this figure, it can be seen that based on the same static features, the full GDESA framework with document selection component is more suitable for those candidate documents with poor initial ranking lists. As we described in Section 7.8, the self-attention network is unaware of the positions in diversified ranking $\mathcal{R}$. Concretely, we can observe that the result of DESA from the first position to the third position keeps zero because the model fails to select and promote the best diversified document. A poor ranking list may only contain a few documents covering intents, and each ranking position must be precisely measured. For the query #36, DESA promotes a document with intent into the range of top-5. However, GDESA has successfully promotes the document to the top-1 ranking position, leading to a better ranking results.

Compared with the DESA variation without document selection, our full GDESA framework can integrate both the global interaction features and the document selection features based on the previous selected documents. With the same static features, the step-by-step sequential selection process of GDESA can precisely model the sequence information at each position in the diversified ranking $\mathcal{R}$, trying to promote the document with more intents to the former ranking positions. These results indicate that the integration of global interaction features and sequential interaction features is beneficial to the overall performance of the framework. For search result diversification task, the global document interaction features are powerful for modeling the interaction between all those candidate documents. However, the document selection process can also contribute to the final results.

While for search result diversification task, the ranking position of each document is of great importance since users usually read the documents from top to bottom in real scenarios. As a result, the sequential interaction is critical for precisely measuring the status of each ranking position, especially the top ranking positions. In conclusion, it is necessary and beneficial to leverage the two kinds of interaction: the global interactions between all documents and the sequential interaction between the selected sequence and each candidate document.

8 CONCLUSION AND FUTURE WORK

In this paper, we propose a search result diversification framework that exploits both global document interaction and document selection. Based on our previous work, we extend the framework with an RNN-based component for document selection. Compared with previous work in diverse ranking tasks, this is the first time that all interactions between candidate documents and between each candidate document and the selected document sequence are modeled. Experimental results show that our framework can significantly outperform all baselines, including document selection models and global interaction models. In comparison to the document selection models, measuring the global document interactions between candidate documents can greatly reduce the gap between the local and global optimal rankings. In comparison to those global interaction models that do not include document selection, it is beneficial to exploit the document selection capability in addition to the global interaction. We also analyze the influence of sequential information in self-attention networks and demonstrate the importance of using an RNN-based document selection component. We demonstrate that, while global document interaction features are powerful for diversification, it is still required and beneficial to integrate the features of document selection. To simplify the problem in this work, we reuse the document relevance features and embeddings from previous works, and one of our future work is using a BERT-based relevance matching approach or document feature extractor to improve the quality of document representation. Besides, our document selection component only considers document novelty, another future work could be integrating subtopic coverage into the document selection component.

ACKNOWLEDGMENTS

Ji-Rong Wen is the corresponding author. This work was supported by National Natural Science Foundation of China No. 61872370 and No. 61832017, Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098, the Fundamental Research Funds for the Central Universities, the Research Funds of Renmin University of China NO. 22XNKJ34, Intelligent Social Governance Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China. We also acknowledge the support provided and contribution made by Public Policy and Decision-making Research Lab of Renmin University of China.

REFERENCES

[1] Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. 2009. Diversifying Search Results. In *Proceedings of the Second International Conference on Web Search and Web Data Mining, WSDM 2009, Barcelona, Spain, February 9-11, 2009*, Ricardo Baeza-Yates, Paolo Boldi, Berthier A. Ribeiro-Neto, and Berkant Barla Cambazoglu (Eds.). ACM, 5–14. https://doi.org/10.1145/1498759.1498766

[2] Qingyao Ai, Keping Bi, Jiafeng Guo, and W. Bruce Croft. 2018. Learning a Deep Listwise Context Model for Ranking Refinement. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, Kevyn Collins-Thompson, Qiaozhu Mei, Brian D. Davison, Yiqun Liu, and Emine Yilmaz (Eds.). ACM, 135–144. https://doi.org/10.1145/3209978.3209985

[3] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer Normalization. *CoRR* abs/1607.06450 (2016). arXiv:1607.06450 http://arxiv.org/abs/1607.06450

[4] Ricardo A. Baeza-Yates, Carlos A. Hurtado, and Marcelo Mendoza. 2004. Query Recommendation Using Query Logs in Search Engines. In *Current Trends in Database Technology - EDBT 2004 Workshops, EDBT 2004 Workshops PhD, DataX, PIM, P2P&DB, and ClustWeb, Heraklion, Crete, Greece, March 14-18, 2004, Revised Selected Papers (Lecture Notes in Computer Science, Vol. 3268)*, Wolfgang Lindner, Marco Mesiti, Can Türker, Yannis Tzitzikas, and Athena Vakali (Eds.). Springer, 588–596. https://doi.org/10.1007/978-3-540-30192-9_58

[5] Jaime G. Carbonell and Jade Goldstein. 1998. The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28 1998, Melbourne, Australia*, W. Bruce Croft, Alistair Moffat, C. J. van Rijsbergen, Ross Wilkinson, and Justin Zobel (Eds.). ACM, 335–336. https://doi.org/10.1145/290941.291025

[6] Olivier Chapelle, Donald Metlzer, Ya Zhang, and Pierre Grinspan. 2009. Expected Reciprocal Rank for Graded Relevance. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*, David Wai-Lok Cheung, Il-Yeol Song, Wesley W. Chu, Xiaohua Hu, and Jimmy Lin (Eds.). ACM, 621–630. https://doi.org/10.1145/1645953.1646033

[7] Charles L. A. Clarke, Nick Craswell, and Ian Soboroff. 2009. Overview of the TREC 2009 Web Track. In *Proceedings of The Eighteenth Text REtrieval Conference, TREC 2009, Gaithersburg, Maryland, USA, November 17-20, 2009 (NIST Special Publication, Vol. 500-278)*, Ellen M. Voorhees and Lori P. Buckland (Eds.). National Institute of Standards and Technology (NIST). http://trec.nist.gov/pubs/trec18/papers/WEB09.OVERVIEW.pdf

[8] Charles L. A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and Diversity in Information Retrieval Evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, Sung-Hyon Myaeng, Douglas W. Oard, Fabrizio Sebastiani, Tat-Seng Chua, and Mun-Kew Leong (Eds.). ACM, 659–666. https://doi.org/10.1145/1390334.1390446

[9] Van Dang and W. Bruce Croft. 2012. Diversity by Proportionality: An Election-based Approach to Search Result Diversification. In *The 35th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '12, Portland, OR, USA, August 12-16, 2012*, William R. Hersh, Jamie Callan, Yoelle Maarek, and Mark Sanderson (Eds.). ACM, 65–74. https://doi.org/10.1145/2348283.2348296

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, 4171–4186. https://doi.org/10.18653/v1/n19-1423

[11] 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, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, Carey L. Williamson, Mary Ellen Zurko, Peter F. Patel-Schneider, and Prashant J. Shenoy (Eds.). ACM, 581–590. https://doi.org/10.1145/1242572.1242651

[12] Yue Feng, Jun Xu, Yanyan Lan, Jiafeng Guo, Wei Zeng, and Xueqi Cheng. 2018. From Greedy Selection to Exploratory Decision-Making: Diverse Ranking with Policy-Value Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, Kevyn Collins-Thompson, Qiaozhu Mei, Brian D. Davison, Yiqun Liu, and Emine Yilmaz (Eds.). ACM, 125–134. https://doi.org/10.1145/3209978.3209979

[13] Sha Hu, Zhicheng Dou, Xiaojie Wang, Tetsuya Sakai, and Ji-Rong Wen. 2015. Search Result Diversification Based on Hierarchical Intents. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, James Bailey, Alistair Moffat, Charu C. Aggarwal, Maarten de Rijke, Ravi Kumar, Vanessa Murdock, Timos K. Sellis, and Jeffrey Xu Yu (Eds.). ACM, 63–72. https://doi.org/10.1145/2806416.2806455

[14] Bernard J. Jansen, Amanda Spink, and Tefko Saracevic. 2000. Real Life, Real Users, and Real Needs: A Study and Analysis of User Queries on the Web. *Inf. Process. Manag.* 36, 2 (2000), 207–227. https://doi.org/10.1016/S0306-4573(99)00056-4

[15] Zhengbao Jiang, Ji-Rong Wen, Zhicheng Dou, Wayne Xin Zhao, Jian-Yun Nie, and Ming Yue. 2017. Learning to Diversify Search Results via Subtopic Attention. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White (Eds.). ACM, 545–554. https://doi.org/10.1145/3077136.3080805

[16] Quoc V. Le and Tomás Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014 (JMLR Workshop and Conference Proceedings, Vol. 32)*. JMLR.org, 1188–1196. http://proceedings.mlr.press/v32/le14.html

[17] Shangsong Liang, Fei Cai, Zhaochun Ren, and Maarten de Rijke. 2016. Efficient Structured Learning for Personalized Diversification. *IEEE Trans. Knowl. Data Eng.* 28, 11 (2016), 2958–2973. https://doi.org/10.1109/TKDE.2016.2594064

[18] Jiongnan Liu, Zhicheng Dou, Xiaojie Wang, Shuqi Lu, and Ji-Rong Wen. 2020. DVGAN: A Minimax Game for Search Result Diversification Combining Explicit and Implicit Features. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 479–488. https://doi.org/10.1145/3397271.3401084

[19] Liang Pang, Qingyao Ai, and Jun Xu. 2021. Beyond Probability Ranking Principle: Modeling the Dependencies among Documents. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 2647–2650. https://doi.org/10.1145/3404835.3462808

[20] Liang Pang, Jun Xu, Qingyao Ai, Yanyan Lan, Xueqi Cheng, and Jirong Wen. 2020. SetRank: Learning a Permutation-Invariant Ranking Model for Information Retrieval. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 499–508. https://doi.org/10.1145/3397271.3401104

[21] Rama Kumar Pasumarthi, Honglei Zhuang, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2020. Permutation Equivariant Document Interaction Network for Neural Learning to Rank. In *ICTIR '20: The 2020 ACM SIGIR International Conference on the Theory of Information Retrieval, Virtual Event, Norway, September 14-17, 2020*, Krisztian Balog, Vinay Setty, Christina Lioma, Yiqun Liu, Min Zhang, and Klaus Berberich (Eds.). ACM, 145–148. https://doi.org/10.1145/3409256.3409819

[22] Xubo Qin, Zhicheng Dou, and Ji-Rong Wen. 2020. Diversifying Search Results using Self-Attention Network. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (Eds.). ACM, 1265–1274. https://doi.org/10.1145/3340531.3411914

[23] Xubo Qin, Zhicheng Dou, Yutao Zhu, and Ji-Rong Wen. 2021. Interaction-Based Document Matching for Implicit Search Result Diversification. In *Information Retrieval - 27th China Conference, CCIR 2021, Dalian, China, October 29-31, 2021, Proceedings (Lecture Notes in Computer Science, Vol. 13026)*, Hongfei Lin, Min Zhang, and Liang Pang (Eds.). Springer, 3–15. https://doi.org/10.1007/978-3-030-88189-4_1

[24] Zhen Qin, Le Yan, Honglei Zhuang, Yi Tay, Rama Kumar Pasumarthi, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2021. Are Neural Rankers still Outperformed by Gradient Boosted Decision Trees?. In *9th International Conference on Learning Representations,*

ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net. https://openreview.net/forum?id=Ut1vF_q_vC

[25] S. E. Robertson. 1997. *The Probability Ranking Principle in IR.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 281–286.

[26] Tetsuya Sakai and Ruihua Song. 2011. Evaluating Diversified Search Results Using Per-intent Graded Relevance. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25-29, 2011*, Wei-Ying Ma, Jian-Yun Nie, Ricardo Baeza-Yates, Tat-Seng Chua, and W. Bruce Croft (Eds.). ACM, 1043–1052. https://doi.org/10.1145/2009916.2010055

[27] Rodrygo L. T. Santos, Craig Macdonald, and Iadh Ounis. 2010. Exploiting Query Reformulations for Web Search Result Diversification. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, Michael Rappa, Paul Jones, Juliana Freire, and Soumen Chakrabarti (Eds.). ACM, 881–890. https://doi.org/10.1145/1772690.1772780

[28] Rodrygo L. T. Santos, Craig MacDonald, and Iadh Ounis. 2015. Search Result Diversification. *Found. Trends Inf. Retr.* 9, 1 (2015), 1–90. https://doi.org/10.1561/1500000040

[29] Craig Silverstein, Monika Rauch Henzinger, Hannes Marais, and Michael Moricz. 1999. Analysis of a Very Large Web Search Engine Query Log. *SIGIR Forum* 33, 1 (1999), 6–12. https://doi.org/10.1145/331403.331405

[30] Ruihua Song, Zhenxiao Luo, Ji-Rong Wen, Yong Yu, and Hsiao-Wuen Hon. 2007. Identifying Ambiguous Queries in Web Search. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, Carey L. Williamson, Mary Ellen Zurko, Peter F. Patel-Schneider, and Prashant J. Shenoy (Eds.). ACM, 1169–1170. https://doi.org/10.1145/1242572.1242749

[31] Zhan Su, Zhicheng Dou, Yutao Zhu, Xubo Qin, and Ji-Rong Wen. 2021. Modeling Intent Graph for Search Result Diversification. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 736–746. https://doi.org/10.1145/3404835.3462872

[32] Chongyang Tao, Wei Wu, Can Xu, Wenpeng Hu, Dongyan Zhao, and Rui Yan. 2019. Multi-Representation Fusion Network for Multi-Turn Response Selection in Retrieval-Based Chatbots. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, J. Shane Culpepper, Alistair Moffat, Paul N. Bennett, and Kristina Lerman (Eds.). ACM, 267–275. https://doi.org/10.1145/3289600.3290985

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[34] Long Xia, Jun Xu, Yanyan Lan, Jiafeng Guo, and Xueqi Cheng. 2015. Learning Maximal Marginal Relevance Model via Directly Optimizing Diversity Evaluation Measures. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13, 2015*, Ricardo Baeza-Yates, Mounia Lalmas, Alistair Moffat, and Berthier A. Ribeiro-Neto (Eds.). ACM, 113–122. https://doi.org/10.1145/2766462.2767710

[35] Long Xia, Jun Xu, Yanyan Lan, Jiafeng Guo, and Xueqi Cheng. 2016. Modeling Document Novelty with Neural Tensor Network for Search Result Diversification. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*, Raffaele Perego, Fabrizio Sebastiani, Javed A. Aslam, Ian Ruthven, and Justin Zobel (Eds.). ACM, 395–404. https://doi.org/10.1145/2911451.2911498

[36] Long Xia, Jun Xu, Yanyan Lan, Jiafeng Guo, Wei Zeng, and Xueqi Cheng. 2017. Adapting Markov Decision Process for Search Result Diversification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White (Eds.). ACM, 535–544. https://doi.org/10.1145/3077136.3080775

[37] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-End Neural Ad-hoc Ranking with Kernel Pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White (Eds.). ACM, 55–64. https://doi.org/10.1145/3077136.3080809

[38] Jun Xu, Zeng Wei, Long Xia, Yanyan Lan, Dawei Yin, Xueqi Cheng, and Ji-Rong Wen. 2020. Reinforcement Learning to Rank with Pairwise Policy Gradient. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 509–518. https://doi.org/10.1145/3397271.3401148

[39] Le Yan, Zhen Qin, Rama Kumar Pasumarthi, Xuanhui Wang, and Michael Bendersky. 2021. Diversification-Aware Learning to Rank using Distributed Representation. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 127–136. https://doi.org/10.1145/3442381.3449831

[40] Sevgi Yigit-Sert, Ismail Sengor Altingovde, Craig Macdonald, Iadh Ounis, and Özgür Ulusoy. 2020. Supervised Approaches for Explicit Search Result Diversification. *Inf. Process. Manag.* 57, 6 (2020), 102356. https://doi.org/10.1016/j.ipm.2020.102356

[41] Yisong Yue and Thorsten Joachims. 2008. Predicting Diverse Subsets Using Structural SVMs. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008 (ACM International Conference Proceeding Series, Vol. 307)*, William W. Cohen, Andrew McCallum, and Sam T. Roweis (Eds.). ACM, 1224–1231. https://doi.org/10.1145/1390156.1390310

[42] ChengXiang Zhai, William W. Cohen, and John D. Lafferty. 2003. Beyond Independent Relevance: Methods and Evaluation Metrics for Subtopic Retrieval. In *SIGIR 2003: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 28 - August 1, 2003, Toronto, Canada*, Charles L. A. Clarke, Gordon V. Cormack, Jamie Callan, David Hawking, and Alan F. Smeaton (Eds.). ACM, 10–17. https://doi.org/10.1145/860435.860440

[43] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, 1441–1451. https://doi.org/10.18653/v1/p19-1139

[44] Xiangyang Zhou, Lu Li, Daxiang Dong, Yi Liu, Ying Chen, Wayne Xin Zhao, Dianhai Yu, and Hua Wu. 2018. Multi-Turn Response Selection for Chatbots with Deep Attention Matching Network. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, 1118–1127. https://doi.org/10.18653/v1/P18-1103

[45] Yadong Zhu, Yanyan Lan, Jiafeng Guo, Xueqi Cheng, and Shuzi Niu. 2014. Learning for Search Result Diversification. In *The 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, Gold Coast , QLD, Australia - July 06 - 11, 2014*, Shlomo Geva, Andrew Trotman, Peter Bruza, Charles L. A. Clarke, and Kalervo Järvelin (Eds.). ACM, 293–302. https://doi.org/10.1145/2600428.2609634

[46] 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*, Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, and Hanghang Tong (Eds.). ACM, 2780–2791. https://doi.org/10.1145/3459637.3482243

[47] Yutao Zhu, Jian-Yun Nie, Kun Zhou, Pan Du, Hao Jiang, and Zhicheng Dou. 2021. Proactive Retrieval-based Chatbots based on Relevant Knowledge and Goals. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 2000–2004. https://doi.org/10.1145/3404835.3463011

[48] Yutao Zhu, Ruihua Song, Zhicheng Dou, Jian-Yun Nie, and Jin Zhou. 2020. ScriptWriter: Narrative-Guided Script Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (Eds.). Association for Computational Linguistics, 8647–8657. https://doi.org/10.18653/v1/2020.acl-main.765

[49] Yutao Zhu, Ruihua Song, Jian-Yun Nie, Pan Du, Zhicheng Dou, and Jin Zhou. 2022. Leveraging Narrative to Generate Movie Script. *ACM Trans. Inf. Syst.* 40, 4, Article 86 (mar 2022), 32 pages. https://doi.org/10.1145/3507356