the model hard to train. So the main challenge in training a supervised diversification model is how to sample enough high-quality training data that contain an appropriate number of relevant documents from the candidate document set. Some methods such as R-LTR [26] only use the top documents in the ideal rankings while other methods such as PAMM [23] sample the training rankings by judging it through diversification evaluation metrics. However, none of the existing approaches solve this problem completely. The quality of training data used by R-LTR [26] is high but its quantity is too small to train the model which may lead to underfit. The quantity of the training dataset used by PAMM [23] is large enough but the quality of it depends on some hyper-parameters such as the range of α-nDCG [4] of negative ranking samples, which may cause the model hard to tune. How to generate training samples effectively is still a challenge for training diversification models.

To tackle this problem, inspired by IRGAN [19], we introduce Generative Adversarial Network (GAN) [10] into search result diversification. Generator generates negative training samples instead of using handcrafted rules for discriminator to train and discriminator provides reward for generator for better sampling. This positive feedback mechanism may improve sampling performance. Furthermore, there are two main components in GAN, so it is natural to use two different approaches in generator and discriminator. This feature of GAN provides a simple way to combine explicit document-subtopic relevance features and implicit document-document similarity features to improve the performance of search result diversification. Specifically, we use explicit approaches in generator and implicit approaches in discriminator in this paper, and it is easy to switch to different settings in real applications. We call this framework DVGAN(search result DiVersification using Generative Adversarial Network). In our framework, we also propose two training methods, namely document selection method DVGAN-doc and ranking selection method DVGAN-rank. Different from the traditional GAN, generator needs input data (for example, the selected document ranking) to generate the negative samples. In our framework, the input data sampling is a part of the data sampling problem. To better generate the input data for better sampling training data, in these two methods, we added a new component sampler to the GAN framework and proposed several sampling algorithms to reduce the high dimension of the sampling space. Experimental results on TREC Web Track data show our methods outperform the existing methods significantly.

The main contribution of this paper is threefold: (1) To the best of our knowledge, this is the first method adapting generative adversarial network to search result diversification;(2) We combine explicit features and implicit features in GAN to improve diversification quality;(3) We proposed several sampling algorithms considering both quality and quantity of the training data to solve the problem of training data sampling.

The rest of the paper is organized as follows. We introduce related works in Section 2. Following this we introduce the GAN framework for search result diversification in Section 3. In Section 4, we elaborate each component of our proposed model. We describe experimental settings in Section 5 and analyze experimental results in Section 6. We conclude the paper in Section 7.

2 RELATED WORK

2.1 Search Result Diversification

As search result diversification is an effective way to solve the problem of query ambiguity, many models have been proposed to solve this problem [1, 6, 12, 17, 18, 23–26]. Depending on whether the subtopics of query are explicitly modeled and the form of score function, existing diversification approaches can be categorized into implicit and explicit approaches. We will introduce these two approaches respectively in the following part.

Implicit Diversification Approaches. The implicit approaches emphasize the document's relevance to the query and novelty to the selected documents. In the early years' research on diversification, implicit methods are most unsupervised. MMR [1] can be regarded as the foundation of implicit methods. Its diversification score function is as follows.

$$f(d|q,S) = (1-\lambda)S^{\text{rel}}(d,q) + \lambda\Lambda_{d_j \in S} S^{\text{div}}(d,d_j). \quad (1)$$

The S^{rel} function reflects the d's relevance to the query q and the S^{div} function reflects the d's novelty to the list of documents S that are already ranked before the current document d. The Λ function is to aggregate the novelty between document d and S. Most implicit methods replace the S^{rel} and S^{div} with more complex function and design loss function to use the machine learning method to improve the performance. The relational learning-to-rank (R-LTR) [26] replaces the S^{div} score by using the relationship matrix between document d and selected documents S. And the loss function is inspired by the learning to rank which is aiming to maximize the probability of optimal rankings. Based on the same score function of R-LTR, Xia et al. proposed PAMM [23] in which loss function is designed to directly maximize the score margin of positive and negative rankings. Furthermore, Neural Tensor Network (NTN) [24] was introduced into search result diversification to measure document similarity automatically. In our framework, we use the score function of the R-LTR in discriminator.

Explicit Diversification Approaches. Different from implicit approaches which mainly model document novelty based on similarities between documents, explicit approaches regard the query as several subtopics, and explicitly leverage subtopics to determine the diversity of results [6, 12, 17, 18]. Most explicit approaches focus on the subtopic coverage of results, by calculating subtopic distribution based on ranked documents. The score function for explicit methods can be described as the following form:

$$f(d|q,S) = (1-\lambda)S^{\text{rel}}(d,q) + \lambda \sum_{i \in I_q} A(i|S) * S^{\text{sub}}(d,i), \quad (2)$$

where i denotes one subtopic of the query subtopics, I_q denotes all the subtopics of query q. $A(i|S)$ denotes the distribution of the subtopic given the selected documents rankings S which contains the document-subtopic information of previous documents. The S^{rel} function reflects the d's relevance to the query q and the S^{sub} function reflects the d's relevance to the subtopics. xQuAD [18] is one of the representative methods of unsupervised explicit approaches. It defines the subtopic distribution by calculating the probability of the selected documents not covering the subtopics. PM2 [6] is another unsupervised explicit method calculating the

distribution by counting the relevant document of the subtopic. DSSA [12] introduces the machine learning method into explicit approaches. It calculates the distribution using the RNN and attention mechanism [16]. In our framework, we mainly use the score function of the DSSA in our generator.

Discussion. As described above, existing implicit and explicit approaches considered different information for learning diversification functions and had different goals. The implicit approaches mainly use the dissimilarity between documents and emphasize the novelty of the documents, whereas the explicit approaches exploit subtopics and stress the coverage. So it is intuitive that using both kinds of features may lead to potential performance improvement compared to the sole use of explicit or implicit approaches. In this paper, we will have a preliminary study on this.

2.2 Generative Adversarial Network

Generative Adversarial Network(GAN) [10] is initially used in the area of computer vision to generate pictures that are similar to realistic. There are two models in the Generative Adversarial Network, which are called generator and discriminator. These two models are trained in an adversarial minimax game. This process aims at erasing the unnecessary noise in the contiguous dataset, which is a semi-supervised model. In recent years, after overcoming the problem of passing gradients from the discriminator to the generator, GAN has just been introduced into a discrete area. For example, SeqGAN [13] introduces the GAN to the text sequence generation area combined with Monte Carlo search. It is also used in the traditional information retrieval area, Wang proposed IRGAN [19] which consists of two information retrieval models in it. Comparing to other information retrieval models, IRGAN's generator can provide negative training samples with higher quality. In the personalized search area, Lu proposed PSGAN [15] inspired by IRGAN. Our framework is inspired by the former two models. However, we combine the idea of minimax game with the method in diversification search. In this paper we will discuss how to apply GAN in our framework to generate more relative and well-covered document ranking answering the query. We also apply GAN to combine explicit and implicit approaches.

3 DVGAN-A GAN FRAMEWORK FOR SEARCH RESULT DIVERSIFICATION

3.1 Problem Formulation and Framework

As we introduced in Section 1, we want to combine both explicit approach and implicit approach to improve the search result diversification performance via generative adversarial network. We will use explicit approach's score function in generator because its form is close to the diversification evaluation metrics. We use implicit approach's score function in discriminator as its form is strong to distinguish the positive samples and negative samples which are closed by directly comparing documents. Through the minimax game training process, we expect that discriminator can provide rewards including the information reflecting implicit features for generator for optimizing the calculation of subtopic distribution in it, and generator can generate high-quality negative samples to discriminator to produce more useful feedback. In this way, the generator can produce a high-quality document ranking as diversification search results.

Table 1: Notations in our framework

Name	Description
Q, q	the query set, a query in the set, $q \in Q$
S	a list of documents that are already ranked
$\mathcal{S}_q$	a set of document list S for query q, $S \in \mathcal{S}_q$
C	a set of candidate documents
$\mathcal{C}_q$	a collection of document sets, $C \in \mathcal{C}_q$
Γ	data given to the generator
Ξ', ξ	set of generated samples, ξ is a sample
$p_{\text{true}}(\xi\|q,\Gamma)$	true distribution of samples
$p_\theta(\xi\|q,\Gamma)$	distribution of generated samples
D', d	set of generated documents, d is a document
L', l	set of generated ranking lists, l is a ranking list
$\mathcal{G}, \mathcal{D}$	generator, discriminator
ϕ, θ	parameters in $\mathcal{G}$ and $\mathcal{D}$
f_θ, f_ϕ	diversification function in $\mathcal{G}$*and* $\mathcal{D}$
$\mathcal{G}_\theta(\xi)$,$\mathcal{D}_\phi(\xi)$	score function of sample ξ in $\mathcal{G}$ and $\mathcal{D}$

The notations used in this paper are listed in Table 1. We propose two different training methods, namely DVGAN-doc and DVGAN-rank. They mainly differ in what ξ the generator generates and input data Γ are given to the generator. The item ξ generated by the generator could be a document d or a ranking $l = \{d_{l_1}, \cdots, d_{l_i}, \cdots, d_{l_{|l|}}\}$ (i.e., ξ is d or l). Γ could be selected document ranking S or candidate document set C (i.e., Γ is S or C). In DVGAN-doc method, generator generates negative document set D' given the query q and selected document ranking S. In DVGAN-rank method, generator generates complete negative document ranking set L' given the query q and candidate document set C. In both methods, there are three components in the adversarial framework: a generator, a discriminator, and a sampler.

Suppose $f_\theta(d|q,S)$ and $f_\phi(d|q,S)$ are the diversification score functions of a document d in generator and discriminator. In DVGAN-doc method, the score function of ξ (particular d) is the same as the diversification score function $f_\theta(d|q,S)$. In DVGAN-rank method, we calculate the score function of ξ (particular l), i.e., $\mathcal{D}_\theta(l|q,C)$ and $\mathcal{G}_\phi(l|q,C)$, using Plackett-Luce model [7]. Specifically, we have:

$$\text{DVGAN-doc} \begin{cases} \mathcal{D}_\phi(d|q,S) = f_\phi(d|q,S) \\ \mathcal{G}_\theta(d|q,S) = f_\theta(d|q,S), \end{cases}$$

$$\text{DVGAN-rank} \begin{cases} \mathcal{D}_\phi(l|q,C) = \prod_{i=1}^{|l|} \frac{f_\phi(d_{l_i}|q,\{d_{l_1},...,d_{l_{i-1}}\})}{\sum_{j=i}^{|l|} f_\phi(d_{l_j}|q,\{d_{l_1},...,d_{l_{i-1}}\})} \\ \mathcal{G}_\theta(l|q,C) = \prod_{i=1}^{|l|} \frac{f_\theta(d_{l_i}|q,\{d_{l_1},...,d_{l_{i-1}}\})}{\sum_{j=i}^{|l|} f_\theta(d_{l_j}|q,\{d_{l_1},...,d_{l_{i-1}}\})} \end{cases} \tag{3}$$

Suppose $p_{\text{true}}(\xi|q,\Gamma)$ is the true distribution of samples which generator tries to fit and $p_\theta(\xi|q,\Gamma)$ is the distribution of generated samples. $p_{\text{true}}(\xi|q,\Gamma) = p_{\text{true}}(d|q,S)$ is the true distribution of documents in DVGAN-doc (i.e., $\Gamma = S$), and $p_{\text{true}}(\xi|q,\Gamma) = p_{\text{true}}(l|q,C)$

is the true distribution of rankings in DVGAN-rank (i.e., $\Gamma = C$). The form of $p_\theta(\xi|q,\Gamma)$ for both methods is the same, and it is calculated by the softmax function:

$$\begin{aligned} p_\theta(\xi|q,\Gamma) &= \frac{\exp(\mathcal{G}_\theta(\xi|q,\Gamma))}{\sum_{\xi}\exp(\mathcal{G}_\theta(\xi|q,\Gamma))} \\ &= \frac{\exp(\mathcal{G}_\theta(\xi|q,\Gamma))}{\sum_{\xi\in\Xi'}\exp(\mathcal{G}_\theta(\xi|q,\Gamma)) + \exp(\mathcal{G}_\theta(\xi_{\text{true}}|q,\Gamma))}, \end{aligned} \tag{4}$$

where ξ_{true} denotes the positive sample in discriminator. In different methods, we can simply replace the ξ,Ξ' by the document d, document set D' or ranking l, ranking set L'.

The discriminator aims to learn a score distribution $\mathcal{D}_\phi(\xi|q,\Gamma)$, which is to distinguish the samples which satisfy the demand of diversification (positive sample) from the generated samples (negative samples). The documents or document rankings with the high metric score are treated as positive samples and what the generator generates are treated as negative samples. According to the Eq. (3), we can infer that two forms of $\mathcal{D}_\phi$ are both related to $f_\phi(d|q,S)$. So the problem is simplified to learn the $f_\phi(d|q,S)$ diversification score function. In our method, $f_\phi(d|q,S)$ is implemented as the implicit diversification score function in order to better distinguish positive and negative samples as the implicit approaches directly model the dissimilarity between documents.

The generator aims to learn a distribution $p_\theta(\xi|q,\Gamma)$ to fit the real distribution $p_{\text{true}}(\xi|q,\Gamma)$ through the score function $\mathcal{G}_\theta(\xi|q,\Gamma)$. The generator also generates negative samples according to the p_θ to confuse the discriminator. According to the Eq. (3), we can infer that two forms of $\mathcal{G}_\phi$ are both related to $f_\theta(d|q,S)$. So the problem is simplified to learn the $f_\theta(d|q,S)$ diversification score function. In our method, $f_\theta(d|q,S)$ is implemented as the explicit diversification score function in order to generate more confusing samples as the explicit approaches directly model the coverage of subtopics.

The sampler aims to generate the input data Γ to generator. In DVGAN-doc, it is a list of documents which are already ranked, i.e., S. In DVGAN-rank, it is a set of candidate documents C. The generator will generate a corresponding object ξ (a document for DVGAN-doc or a ranking list for DVGAN-rank) based on Γ, hence the sampler is also a critical component that helps reduce the sampling space.

The minimax game in DVGAN framework can be described as follows: given a query q, its subtopics, and the input data Γ given by the sampler, the generator tries to generate the best samples set Ξ' (set of documents D' in DVGAN-doc and set of ranking lists L'in DVGAN-rank) that satisfies the diversification demand with high relevance to the query and coverage of the subtopics. The discriminator tries to distinguish the true document or rankings ξ_{true} from the negative samples ξ_θ generated by the generator. Formally, given query set Q and document, we have:

$$J^{\mathcal{G}*,\mathcal{D}*} = \min_\theta \max_\phi \mathfrak{I}(p_{\text{true}}, p_\theta), \tag{5}$$

We will introduce the specific form of $\mathfrak{I}$ in DVGAN-doc and DVGAN-rank respectively in the following part.

3.2 DVGAN-doc: Document Selection Method

DVGAN-doc is a natural extension of IRGAN considering diversification features. Generator tries to select documents that resemble the positive documents from the candidate document set to fool the discriminator, whereas discriminator tries to distinguish the positive and negative documents. The $\mathfrak{I}$ in Eq. (5) in DVGAN-doc is as follows:

$$\begin{aligned} \mathfrak{I}(p_{\text{true}}, p_\theta) = \sum_{q\in Q, S\in\mathcal{S}_q} \Big(&\mathbb{E}_{d\sim p_{\text{true}}(d|q,S)} \log D_\phi(d|q,S) \\ &+ \mathbb{E}_{d\sim p_\theta(d|q,S)} \log\left(1 - D_\phi(d|q,S)\right)\Big), \end{aligned} \tag{6}$$

where generator G is written as $p_\theta(d|q,S)$. Eq. (4) and the discriminator D is the estimated probability calculated by:

$$D_\phi(d|q,S) = \sigma\left(f_\phi(d|q,S)\right) = \frac{\exp(f_\phi(d|q,S))}{1+\exp(f_\phi(d|q,S))}. \tag{7}$$

Please note that different from IRGAN [19], DVGAN-doc has an additional component S to represent the former selected documents. As S contain order information, it is required to be ranked.

3.2.1 Optimizing Discriminator. According to the Eq. (6) optimizing the discriminator is to optimize ϕ to maximize the whole result given the true documents and generated documents, i.e.,

$$\begin{aligned} \phi^* = \arg\max_\phi \sum_{q\in Q, S\in S_q} (&\mathbb{E}_{d\sim p_{\text{true}}(d|q,S)} \log D_\phi(d|q,S) + \\ &\mathbb{E}_{d\sim p_\theta(d|q,S)} \log(1 - D_\phi(d|q,S))). \end{aligned} \tag{8}$$

3.2.2 Optimizing Generator. As GAN is put into practice in the contiguous area firstly, it is difficult to calculate the generator gradient due to its discrete nature. Inspired by IRGAN [19], we generate negative document set D' by selecting the documents from the candidate document set with the highest scores. Formally, the gradient of the generator is:

$$\nabla_\theta J^{\mathcal{G}}(q,S) \simeq \frac{1}{|D'|} \sum_{d\in D'} \nabla_\theta \log p_\theta(d|q,S) \log(1+\exp(f_\phi(d|q,S))). \tag{9}$$

Note that the feedback from the discriminator $\log(1+\exp(f_\phi(d|q,S)))$ can be regarded as the reward to the generator according to the reinforcement learning which contains the implicit information calculated in discriminator.

3.2.3 Sampler. In the DVGAN-doc method, the sampler sample a list of selected documents S which is sent to the generator. As we mentioned in Section 1, training dataset sampling is the main challenge in search result diversification. In this method, we design two methods for sampling. Firstly, it is necessary to sample the ideal ranking but it is not enough. Recall that our method is not an ideal method, so it may produce some documents which may not be as good as the ideal ones. Hence we also sample in the second way: we randomly select some documents from the document set to form the selected document ranking S. However, recall that in the DVGAN-doc method, we require that S is ranked. So the sampler also needs to re-rank S by the diversification metric like α-nDCG [4]. In practice, half of the selected document ranking S is sampled from the ideal ranking, and the other half of the S is sampled in the second random way. In both ways, the positive document is the best next document maximizing α-nDCG given S.The sampling method is described in Algorithm 1.

Algorithm 1 Sampling algorithm used by DVGAN-doc Sampler

input: query set Q, document set D_q and ideal document ranking r_q for each query q, number of random sample n_s, number of selected documents in random sample d_s.
output: set of selected ranking lists $\mathcal{S}_q$ for each query q.
for query $q \in Q$ **do**
  $\mathcal{S}_q \leftarrow \emptyset$ //the first method, sampling from ideal ranking
  $l \leftarrow \text{len}(r_q)$
  for i = 1 to l **do**
    $\mathcal{S}_q \leftarrow \mathcal{S}_q \cup r_q[:i]$
  end for
  //the second method, random sampling
  for i = 1 to n_s **do**
    $S \leftarrow \emptyset$
    for j = 1 to d_s **do**
      $S \leftarrow S \cup \text{random\_in}(D_q)$
    end for
    $S \leftarrow \text{re} - \text{rank(S)}$
    $\mathcal{S}_q \leftarrow \mathcal{S}_q \cup S$
  end for
end for
return $\mathcal{S}_q$

3.3 DVGAN-rank: Ranking Selection Method

In the former DVGAN-doc method, the difference between the positive samples and negative samples is only one document which may not be enough for discriminator to learn. Thus, we put forward the DVGAN-rank method to differ the positive and negative samples at ranking level.

The $\Im$ in the Eq. (5) in DVGAN-rank method is:

$$\Im\left(p_{\text{true}}, p_\theta\right) = \sum_{q\in Q, C\in C_q} \Big(\mathbb{E}_{l^+\sim p_{\text{true}}(l|q,C), l^-\sim p_\theta(l|q,C)} \left[\mathcal{D}_\phi(l^+|q,C) - \mathcal{D}_\phi(l^-|q,C)\right) \le E(l^+|q,C) - E(l^-|q,C)]\Big), \tag{10}$$

where the generator $\mathcal{G}$ is written as $p_\theta(l|q,C)$ and $\mathcal{D}_\phi$ is the diversification score for a whole document ranking in discriminator calculated by Eq. (4) using Plackett-Luce model and the E is the diversification metrics such as α-NDCG and ERR-IA [2]. The form of $\Im$ is inspired by PAMM [23] method which aims to maximize the margin between positive and negative rankings instead of directly judging the rankings respectively.

3.3.1 Optimizing Discriminator. According to the Eq. (10), optimizing the discriminator is to optimize ϕ to maximize the whole result given the true rankings and generated rankings, i.e.,

$$\phi^* = \arg\max_\phi \sum_{q\in Q, C\in C_q} \Big(\mathbb{E}_{l^+\sim p_{\text{true}}(l^+|q,C), l^-\sim p_\theta(l^-|q,C)} \left[\mathcal{D}_\phi(l^+|q,C) - \mathcal{D}_\phi(l^-|q,C)\right) \le E(l^+|q,C) - E(l^-|q,C)]\Big). \tag{11}$$

The loss function of the discriminator is inspired by PAMM [23] method which is aiming to maximize the margin between the positive and negative rankings.

3.3.2 Optimizing Generator. Similar to the DVGAN-doc method, the gradient of generator is also calculated by sampling technique. We select the rankings with the highest scores to form the negative ranking set L'. Formally, the gradient is:

$$\nabla_\theta J^{\mathcal{G}}(q,C) \simeq \frac{1}{|L'|} \sum_{l\in L'} \nabla_\theta \log p_\theta(l|q,C) \log\Big(1 + \exp(\mathcal{D}_\phi(l|q,C))\Big). \tag{12}$$

The feedback from the discriminator $\log(1+\exp(d_\phi(l|q,C)))$ can be described as the reward to the generator according to the reinforcement learning. And the generator is written as $p_\theta(l|q,C)$ and is calculated by Eq. (3). As $\mathcal{G}_\theta$, $\mathcal{D}_\phi$ is a multiply function, the score may be extremely small, so we use normalization and clipping technique in practice.

3.3.3 Sampler. Similar to the sampler in the DVGAN-doc method, the sampler in DVGAN-rank method also needs to generate the candidate document set C. The procedure is simple: we randomly select documents from the document set and then put it together to form the candidate document set C for generator to rank it. But there is another problem with our DVGAN-rank method. From the Eq. (12), we notice that the generator needs to generate several rankings l. However, the way diversification method generates a ranking is as follows:

1. Initially we make the selected ranking S as an empty set Ø.
2. Select the document d with the highest score given S and q.
3. Add the document d to the end S and back to step 2 or stop and output S as the search result if the length of S is enough.

In this process, generator can generate only one ranking. In optimizing generator as Eq. (12), it needs to generate several l to form the negative rankings set L' to calculate the gradient, which is difficult to implement in programming. To solve this problem, we make the sampler do the work of generating negative ranking list as we mentioned before.

DVGAN-rank sampler receives the selected document set C from the sampler and re-ranks it by diversification metrics to get the positive list l^+. To generate the negative samples l^-, we shuffle the positive list by swapping the former and latter document randomly in a hyper-parameter k times. Due to the fact that the quality of the positive rankings l^+ and negative rankings l^- has great effect on the performance as it does in the PAMM method, the performance of DVGAN-rank highly depends on how the selected document set C is sampled and how the negative rankings are shuffled. Therefore, the training dataset sampling still remains a problem in DVGAN-rank which we will try to solve in future work.

The sampling method is described as Algorithm 2. Since we introduced the DVGAN-rank sampler, we can review the former DVGAN-rank method. It is easy to find that in fact the discriminator is just the PAMM method and what the generator does is to score the document rankings DVGAN-rank sampler sampled. Thus it also makes a great challenge for DVGAN-rank sampler to sample the proper document rankings which can imitate the generator. As a result, DVGAN-rank method is hard to tune.

So far we have introduced two different DVGAN methods. In the next section, we will introduce specific forms of the two diversification score function $f_\phi(d|q,S)$ and $f_\theta(d|q,S)$.

Algorithm 2 Sampling algorithm used by DVGAN-rank

input: query set Q, document set D_q for each query q, number of sample n_s, number of candidate documents in a sample d_c and number of negative ranking of a sample n_l
output: the candidate document ranking set C_q for each query q and positive and negative ranking set l_c^+ and l_c^- for each sample document set C
for query $q \in Q$ **do**
  $C_q \leftarrow \emptyset$
  for i = 1 to n_s **do**
    $C \leftarrow \emptyset$
    for j = 1 to d_c **do**
      $C \leftarrow C \cup \text{random\_in}(\text{D}_\text{q})$
    end for
    $C_q \leftarrow C_q \cup C$
  end for
  for sample $C \in C_q$ **do**
    $l_c^+ \leftarrow \text{re} - \text{rank}(\text{C})$
    for i = 1 to n_l **do**
      $l_c^- \leftarrow \text{shuffle}(l_c^+)$
    end for
  end for
end for
return C_q, l_c^+, l_c^-

4 DIVERSIFICATION USING DVGAN

In this section, we instantiate DVGAN-doc and DVGAN-rank to a concrete form and articulate the training algorithms. The main idea of DVGAN is to introduce GAN into diversification to improve the quality of the training dataset. Furthermore, to use more information, we use explicit approach's score function in generator and implicit approach's score function in discriminator. We will introduce the diversification score functions in discriminator and generator in this section.

4.1 The Discriminator

As we mentioned in Section 3, the discriminator tries to distinguish the positive document or document rankings from the negative ones. Thus, the discriminator needs a score function to calculate score for document d given the query q and selected document ranking S. In our method, we adapt the R-LTR [26] score function for discriminator. We introduce the score function $f_\phi(d_i|q, S)$ in the form of Eq. (1):

$$\begin{aligned} f_\phi(d_i|q,S) &= S^{\text{rel}}(d_i,q) + \Lambda_{d_j \in S} S^{\text{div}}(d_i,d_j), \\ S^{\text{rel}}(d_i,q) &= w_r^T(d) x_{d_i,q}, \\ S^{\text{div}}(d_i,d_j) &= R_{ij}, \\ \Lambda &= w_d^T(d) h(R_i,S), \end{aligned} \tag{13}$$

where $x_{d_i,q}$ denotes the relevance feature vector of the document d_i and query q. R_i denotes the relationship matrix between the document d_i and document d_j in the selected document list S. R_{ij} denotes the relationship vector between d_i and d_j. h denotes the relational function. $w_r(d)$ and $w_d(d)$ denotes the parameters in discriminator. The vector R_{ij} usually captures different diversity features which store the implicit information. Here we notice that in discriminator, the model judges the document's diversity by calculating its diversity features with the selected ones and these features are helpful for distinguishing positive and negative samples. The relational function $h(R_i, S)$ is to aggregate the diversity features between the current document d_i and selected documents, which is usually defined in three ways: max, min, and average. In our method, the max way gets the best performance, i.e.,

$$h(R_i,S) = (\max_{d_j \in S} R_{ij1}, ..., \max_{d_j \in S} R_{ijk}).$$

As relevance features $x_{d_i,q}$ are necessary in information retrieval to model the relevance between document and query, the definition of the relationship vector $R_{i,j}$ is crucial to the performance for the R-LTR. Imagine when we compare two documents, we usually compare their titles, texts , etc., which means that we capture their features using several different components. In our method, we use four different diversity features to construct the relationship vector as we show in Table 2.

Table 2: Diversity features for R-LTR

Name	Description
subtopic diversity	euclidean distance based on SVD model
text diversity	cosine-based distance on term vector
title diversity	text diversity on title
anchor text diversity	text diversity on anchor

Subtopic Diversity: Here the subtopic is different from the subtopic in explicit method, we construct the subtopic information from the documents instead of the queries. We use SVD to capture the implicit subtopics of the documents and euclidean distance based on it to calculate the dissimilarity between two documents. We define the subtopic diversity feature as follows:

$$R_{ij1} = \sqrt{\sum_{k=1}^{m} (p(z_k|d_i) - p(z_k|d_j))^2}.$$

Text Diversity: The dissimilarity of text is also useful for diversification. Here we use the traditional $tf * idf$ vector to calculate the cosine distance to represent text diversity, i.e.,

$$R_{ij2} = 1 - \frac{t_i \cdot t_j}{||t_i|| \cdot ||t_j||},$$

where t_i, t_j denotes the weighted document vectors based on the traditional TF-IDF model.

Title Diversity Title is the precise and brief abstract of the document which contains lots of information. The way of computing title diversity is similar to the text diversity.

Anchor Diversity Anchor can precisely describe the content of a document. The way of computing title diversity is similar to that of the text diversity using the text in anchor.

4.2 The Generator

As we mentioned in Section 3, the generator tries to select high-quality documents from the available documents based on the query q and the selected document ranking S or the candidate document C to fool the discriminator. So it also needs a score function. In our

method we adapt the DSSA [12] score function for the generator. We introduce the score function in the form of Eq. (2):

$$\begin{aligned} f_\theta(d_t|q,S) &= (1-\lambda)S^{\text{rel}}(d_t,q) + \lambda \sum_{i\in I_q} A(i|S) * S^{\text{sub}}(d_t,i), \\ S^{\text{rel}}(d_t,q) &= \mathcal{S}(e_{d_t},e_q) + w_r^T(g) * x_{d_t,q}, \\ S^{\text{sub}}(d_t,i_k) &= \mathcal{S}(e_{d_t},e_{i_k}) + w_r^T(g) * x_{d_t,i_k}, \end{aligned} \tag{14}$$

where $x_{d_t,q}, x_{d_t,i_l}$ denotes the relevance feature vectors between document d_t and query q or subtopic i_k. e_{d_t}, e_q, and e_{i_k} denotes the embedding vectors for document d_t ,query q and subtopic i_k. $w_r(g)$ denotes parameters in generator. S^{rel} and S^{sub} both use the $\mathcal{S}$ function to calculate the similarity between document and query or subtopic based on the embedding vector:

$$\mathcal{S}(e_d,e_q) = e_d^T * w_s(g) * e_q, \tag{15}$$

where $w_s(g)$ denotes parameters in generator. Noticed that the subtopic distribution is actually the most important component in explicit approach. The part of calculating the relevance between documents and queries or subtopics is easy to understand. As DSSA, we will introduce the distribution of subtopic $A(i|S)$. DSSA uses both RNN and attention [16] mechanism to calculate it.

Noticed that the selected document ranking S contains order information, it is natural to use RNN to encode the previous document information. We denote that the documents in S is $d_1, \cdots, d_{t-1}$ in convenience of representation. In spite of the kinds of RNN(in our model, we use LSTM [8]), we use H to denote the RNN cell and h_t to denote the hidden state of RNN, which stores the information of previous t documents. Thus the previous document information at t-th position can be derived from the $t-1$-th position and document embedding vector e_{d_t} using RNN method, i.e.,

$$h_t = H(h_{t-1}, e_{d_t}).$$

Thus we get the document information h_{t-1} given selected document ranking S. Similar to the $\mathcal{S}$ function, we calculate the similarity between the document information and subtopic:

$$A'(h_{t-1}, e_{i_k}) = h_{t-1}^T w_a(g) e_{i_k}, \tag{16}$$

where $w_a(g)$ denotes parameters in generator.

In the way above, we mainly use the distributed embedding representation, which may not be effective and accurate, especially under limited data. So we further use relevance feature vector to improve the subtopic distribution calculation. The following equation can be regarded as the max-pooling:

$$\begin{aligned} &A''(x_{d_1,i_k}, \cdots, x_{d_{t-1},i_k}) = \\ &\max\left([w_p^T(g)x_{d_1,i_k}, \cdots, w_p^T(g)x_{d_{t-1},i_k}]\right), \end{aligned}$$

where $w_p(g)$ denotes parameters in generator.

We directly adapt an addictive way to aggregate the two subtopic distribution and then use softmax function to normalize to get the final distribution:

$$\begin{aligned} a_{i_j|S} &= A'(h_{t-1}, e_{i_j}) + A''(x_{d_1,i_j}, \cdots, x_{d_{t-1},i_k}), \\ A(i_j|S) &= \frac{\exp(a_{i_j})}{\sum_{k=1}^{K}\exp(a_{i_k})}, \end{aligned}$$

where K denotes the number of subtopics of query q.

4.3 Feature Vector

In this part, we will briefly introduce some feature vectors we used.

e_d: Embedding vector for document d, which is the distributed representation of document. It can be constructed in different ways, In this paper, we use doc2vec [14] to get document embeddings.

$x_{d,q}$ and $x_{d,i}$: Relevance feature vectors between the document d and query q and subtopic i. We adapt some traditional IR features such as $tf * idf$ and $BM25$ to construct the relevance features.

e_q and e_i: Embedding vectors for query q and subtopic i. It is obvious that the text of query or subtopic is too short to calculate distributed representation in doc2vec method. To solve this problem, we firstly retrieve W documents using the text of subtopic or query by basic retrieval model (such as BM25). Then we concatenated them to form a pseudo document to calculate the corresponding embedding vector via the doc2vec method.

4.4 Training

It is easy to use DVGAN to generate the diversified search result as we show in the former section. We use the generator as the model to generate the final document ranking result.

In the training process, we first train R-LTR [26] and DSSA [12] respectively using MLE loss in both ways. It is because our framework needs a warm start to avoid the deviation in the training process. Then we train them by DVGAN-doc or DVGAN-rank respectively to get the corresponding model.

In implementation of the models, as the list score $\mathcal{G}_\theta(l|q,C)$ and $\mathcal{D}_\phi(l|q,C)$ and contains a successive multiplication as shown in Eq. (3), it may be extremely small and its gradient after a log function may be extremely huge. So we use clipping technique to control the training process of both positive and negative rankings.

4.5 Review of Our Models

Our model attempts to solve the lack of high-quality sampled training data problems in search result diversification by introducing the generative adversarial network. In order to combine explicit and implicit information to improve the performance, the generator and discriminator in the DVGAN framework use the explicit and implicit features respectively. In the training process, the generator can receive implicit information which it cannot obtain from the discriminator via the reward and the discriminator can receive samples in high-quality from the generator. Inspired by IRGAN, we convert the document generation into document selection. The DVGAN-doc is a natural extension of IRGAN, which adds the selected document ranking S into the score function. The DVGAN-rank method combines loss function of the GAN loss and the PAMM method's loss, which converts maximizing the likelihood estimation into the margin between positive and negative rankings.

5 EXPERIMENTAL SETTINGS

5.1 Data Collection

We experiment with the Web Track dataset [11] from 2009 to 2012. There are 198 queries (2 queries are dropped because they have no subtopic judgment) in this dataset. There are 3 to 8 subtopics for each query. The relevance rating is given at subtopic level. We use google query suggestions as subtopics, which are released by Hu

et al. [9] on their website[1]. we only use the first level subtopics and will adapt the hierarchical structure in future work. The weights of these subtopics are assumed to be uniform.

5.2 Evaluation Metrics

Among all the evaluation metrics [2–4, 20, 21], we use ERR-IA [2], α-NDCG [4], and NRBP [3] as our diversity evaluation metrics. They measure the document ranking by calculating the coverage of each subtopic of the query. Consistent with existing work and TREC Web Track, all these metrics are computed on top 20 results of a ranking. We use two-tailed paired t-test to conduct significance testing with p-value < 0.05.

5.3 Baseline Models

We compare DVGAN with several existing diversification methods. We use Lemur as our non-diversified baseline method. We use xQuAD, TxQuAD, HxQuAD [18], PM2 [6], TPM2 [5], and HPM2 [9] as our unsupervised baseline methods. We use ListMLE [22], R-LTR [26], PAMM [6], R-LTR-NTN, PAMM-NTN [24], and DSSA [12] as supervised baseline methods. Top 20 results of Lemur are used to train the supervised methods. Top 50(Z) results of Lemur are used for diversity re-ranking. To generate enough data for the sampler to sample the selected document ranking S or candidate document set C, we use top 100 results returned by Lemur as the sampler input. In order to prove that the combination of explicit and implicit information is effective, we design a simple method using explicit and implicit features called DSSA+R-LTR, which also uses the whole results of Lemur to train. We use 5-fold cross validation to tune the parameters in all experiments based on α-nDCG@20 [4]. A brief introduction to these baselines is as follows.

Lemur. We use the non-diversified results as our baseline. They are produced by the Indri engine based on the Lemur[2].

ListMLE. ListMLE is a learning-to-rank method, which is similar to the R-LTR method without considering diversity.

xQuAD, TxQuAD, HxQuAD, PM2, TPM2, and HPM2. These methods are the representative unsupervised explicit methods whose diversification score functions are similar to that of the Eq. (1). HxQuAD and HPM2 use the hierarchical structure by adding new parameters. These methods require prior relevance rankings to fulfill the re-ranking. In our experiment, we use ListMLE.

R-LTR,PAMM, and NTN. These methods are the representative supervised implicit methods. For the diversity feature, we use the same four features in Table 2 with two more features: link-based diversity and URL-based diversity in [26], for PAMM, we use α-nDCG@20 as the optimization metrics and tune the number of positive rankings l^+ and negative rankings l^- per query. We tune the function $h_S(R)$ from minimal, maximal, and average for the best performance. The feature vector is the same as DVGAN. We optimize NTN based on both R-LTR and PAMM, denoted as R-LTR-NTN and PAMM-NTN respectively. For these two methods, the number of tensor slices is tuned from 1 to 10.

DSSA. DSSA is the supervised explicit method. We use LSTM [8] as the RNN cell for comparison. In our experiments, we conduct the list-pairwise loss [12] to train DSSA method. The feature vector is the same as DVGAN. The result of DSSA(pre-train) in Table 4 is the result of the DSSA model after the pre-train process.

[1] http://playbigdata.ruc.edu.cn/dou/hdiv/

[2] Lemur service: http://boston.lti.cs.cmu.edu/Services/clueweb09 batch/

Table 3: Relevance features for both R-LTR and DSSA

Name	Description	#Features
TF-IDF	the TF-IDF model	5
BM25	BM25 with default parameters	5
LMIR	LMIR with Dirichlet smoothing	5
PageRank	PageRank score	1
#inlinks	number of inlinks	1
#outlinks	number of outlinks	1

DSSA + R-LTR. This method is a linear combination of explicit and implicit approach. The diversification score function of this method is:

$$f(d|q,S) = (1-\lambda-\mu)S^{\text{rel}}(d,q) + \lambda\Lambda_{d_j\in S}S^{\text{div}}(d_i,d) + \mu\sum_{i\in I_q}A(i|S) * S^{\text{sub}}(d,i). \quad (17)$$

The function $S^{\text{rel}}, S^{\text{div}}$, and Λ are the same as R-LTR, S^{sub} is the same as DSSA, the subtopic distribution $A(i)$ is the same as DSSA.

DVGAN. We train DVGAN in two methods respectively to get DVGAN-doc and DVGAN-rank. We use generator as the model to diversify search results. For the feature vector, the 18-dimension relevance feature vector $x_{d,q}$ is listed in Table 3. e_d is the embedding vector via doc2vec method. The query and subtopic embedding vectors e_q, e_i are constructed by the top 20 (W) documents' distribution representation. For the DVGAN-doc method, we sampled about 40 samples of 20-document-length using the random sampling way for each query. For the DVGAN-rank, we sampled about 10 samples of 10 negative rankings of 20-document-length for each query. Besides, we also show the results using top 100(Z) results of Lemur for diversity re-ranking. The results using top 50 documents are shown as DVGAN-doc(50). The results using top 100 documents are shown as DVGAN-doc(100).

For all of the supervised methods, we tune the learning rate r from 10^{-7} to 10^{-1}.

6 EXPERIMENTAL RESULTS

6.1 Overall Results

The overall results are shown in Table 4. We find both DVGAN-doc methods outperform all explicit and implicit baselines including the naive method DSSA + R-LTR and DVGAN-rank methods outperforms all except DSSA. We find that:

(1) The relative improvement over DSSA, the best explicit method, is up to 2.0% in terms of α-nDCG for $Z = 50$ and up to 3.5% for $Z = 100$. The relative improvement over PAMM-NTN, the best implicit method, is up to 11.5% in terms of α-nDCG and up to 13.2% for $Z = 100$. These results show the advantage of combining the explicit and implicit approaches using DVGAN instead of using only one kind of features.

(2) The relative improvement over DSSA + R-LTR method is up to 8.1% in terms of α-nDCG for $Z = 50$ and up to 9.8% for $Z = 100$. This comparison shows that the naive way of combining explicit

Table 4: Performance comparison of all methods. The best result is in bold. † indicates significant improvement over the pre-train model and all baselines except DSSA with p-value<0.05. ⋆ indicates significant improvement over the pre-train model and all baselines with p-value<0.05.

Methods	ERR-IA	α-nDCG	NRBP
Lemur	.271	.369	.232
ListMLE	.287	.387	.249
xQuAD	.317	.413	.284
TxQuAD	.308	.410	.272
HxQuAD	.326	.421	.294
PM2	.306	.411	.267
TPM2	.291	.399	.250
HPM2	.317	.420	.279
R-LTR	.303	.403	.267
PAMM	.309	.411	.271
R-LTR-NTN	.312	.415	.275
PAMM-NTN	.311	.417	.272
DSSA+R-LTR	.328	.430	.302
DSSA	.356	.456	.326
DSSA(pre-train)(50)	.339	.441	.304
DVGAN-rank(50)	.340	.442	.303
DVGAN-doc(50)	.367†	.465†	.334†
DSSA(pre-train)(100)	.342	.446	.306
DVGAN-rank(100)	.343	.448	.305
DVGAN-doc(100)	**.369**†	**.472***	**.334**†

Table 5: Performance for DVGAN-doc with different score function in generators and discriminators. (50/100) indicates the number of documents used for re-ranking(Z)

Gen, Dis	ERR-IA (50/100)	α-nDCG (50/100)	NRBP (50/100)
DSSA, R-LTR	.367/.369	.465/.472	.334/.334
DSSA, DSSA	.355/.363	.455/.465	.332/.330
R-LTR, R-LTR	.341/.353	.441/.454	.305/.318
R-LTR, DSSA	.336/.349	.437/.452	.298/.313

"Gen, Dis" infers "Generator, Discriminator"

and implicit approaches is not effective. One possible reason for the bad performance of the naive method is that the diversification score functions of two different methods DSSA and R-LTR are not on the same scale, which may cause the learning process slow and easy to trap into local minimal.

(3) The relative improvement over the pre-train model(DSSA(pre-train)) is up to 5.4% in terms of α-nDCG for $Z = 50$ and up to 5.8% for $Z = 100$. This comparison shows the great advantage of using loss function of generative adversarial network instead of traditional loss function such as MLE, PAMM and list-pairwise loss in DSSA.

Table 6: Different sampling strategies in DVGAN-doc. (50/100) indicates the number of documents used for re-ranking(Z)

Sampling Strategy	ERR-IA (50/100)	α-nDCG (50/100)	NRBP (50/100)
Ideal sampling	.355/.360	.457/.468	.321/.326
Random sampling	.353/.352	.454/.458	.319/.316
Both	.367/.369	.465/.472	.334/.334

6.2 Effects of Different Generators and Discriminators

In this part, we compare the different configuraions of generator and discriminator and the result is shown in Table 5. Firstly we can infer that using DSSA's score function in generator and R-LTR's score function in discriminator is the best configuration to introduce generative adversarial network into search result diversification as we expected. Only considering the generator, we can infer that using DSSA's score function outperforms using R-LTR's. The reason is that as we use generator to diversify search results, DSSA's score function considering subtopic coverage is close to diversification evaluation metrics such as α-nDCG. Only considering the discriminator, we can infer that using R-LTR's score function outperforms using DSSA's, The reason is that R-LTR's score function directly modeling dissimilarity between documents is useful in distinguishing negative and positive samples that are closed. Thus, discriminator can provide better rewards for generators to improve the performance of diversification. We also do the same study on $Z = 100$, the result is the same.

6.3 Sampling Study in DVGAN

In this part, we compare different strategies of sampling in our DVGAN-doc method and the result is shown in Table 6. The first way is only sampled by the ideal sampling algorithm which is to select the first k documents in the ideal ranking as our selected document list. The second way is random sampling algorithm which is described in Algorithm 1. The result shows that combining both sampling algorithms is better than using only one. The reason is that (1). ideal sampling is helpful for discriminator to distinguish the positive and negative rankings but makes it hard for generator to imitate the real distribution of data because it is too ideal (2). random sampling makes it easy to imitate for generator but can be confusing for discriminator to distinguish the positive and negative rankings and may provide wrong rewards for generator. The reason why the ideal sampling outperforms the random sampling is that the quality of randomly sampled data is under no guarantee and only using it may cause deviation in training. We also do the same study on $Z = 100$, the result is the same.

7 CONCLUSIONS

In this paper, we proposed DVGAN - a framework for search result diversification adversarial training combining both explicit and implicit information to improve the diversification performance and to solve the problem that high-quality dataset is hard to capture. We

proposed two methods in this framework, DVGAN-doc method and DVGAN-rank method. The DVGAN-doc method is a natural and effective extension of IRGAN. The DVGAN-rank is a combination of PAMM loss function and generative adversarial network. We also proposed several sampling algorithms for the input data to generator to better solve the problem of the lack of high-quality training data. During the the training process, discriminator can provide the implicit information which cannot be obtained in generator via the reward for generator and generator can provide more useful negative samples for discriminator. Experimental results confirm the effectiveness of the proposed methods. The adaption of generative adversarial network also solves the problem of the lack of high-quality data in training process. In future work, we plan to improve the DVGAN-rank and to adapt the hierarchical structure to our method.

ACKNOWLEDGMENTS

Zhicheng Dou is the corresponding author. This work was supported by National Key R&D Program of China No. 2018YFC0830703, National Natural Science Foundation of China No. 61872370 and No. 61832017, Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098.

REFERENCES

[1] Jaime Carbonell and Jade Goldstein. 1998. The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98)*. Association for Computing Machinery, New York, NY, USA, 335–336. https://doi.org/10.1145/290941.291025

[2] 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 '09)*. Association for Computing Machinery, New York, NY, USA, 621–630. https://doi.org/10.1145/1645953.1646033

[3] Maheedhar Kolla Charles L. A. Clarke and Olga Vechtomova. 2009. An Effectiveness Measure for Ambiguous and Underspecified Queries. In *Proceedings of the 2nd International Conference on Theory of Information Retrieval: Advances in Information Retrieval Theory, ICTIR 2009*. 188–199. https://doi.org/10.1007/978-3-642-04417-5_17

[4] 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 '08)*. Association for Computing Machinery, New York, NY, USA, 659–666. https://doi.org/10.1145/1390334.1390446

[5] Charles L. Clarke, Maheedhar Kolla, and Olga Vechtomova. 2009. An Effectiveness Measure for Ambiguous and Underspecified Queries. In *Proceedings of the 2nd International Conference on Theory of Information Retrieval: Advances in Information Retrieval Theory (ICTIR '09)*. Springer-Verlag, Berlin, Heidelberg, 188–199. https://doi.org/10.1007/978-3-642-04417-5_17

[6] Van Dang and W. Bruce Croft. 2012. Diversity by Proportionality: An Election-Based Approach to Search Result Diversification. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '12)*. Association for Computing Machinery, New York, NY, USA, 65–74. https://doi.org/10.1145/2348283.2348296

[7] David J. Groggel. 1996. Analyzing and Modeling Rank Data. *Technometrics* 38, 4 (1996), 403–403. https://doi.org/10.1080/00401706.1996.10484555 arXiv:https://www.tandfonline.com/doi/pdf/10.1080/00401706.1996.10484555

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[9] 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 on Conference on Information and Knowledge Management (CIKM '15)*. Association for Computing Machinery, New York, NY, USA, 63–72. https://doi.org/10.1145/2806416.2806455

[10] Mehdi Mirza Bing Xu David Warde-Farley Sherjil Ozair Aaron Courville Ian J. Goodfellow, Jean Pouget-Abadie and Yoshua Bengio. 2014. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, NIPS 2014*. 2672–2680. https://doi.org/10.5555/2969033.2969125

[11] Changkuk Yoo Jamie Callan, Mark Hoy and Le Zhao. 2009. *Clueweb09 data set*. https://boston.lti.cs.cmu.edu/Data/clueweb09/

[12] 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 (SIGIR '17)*. Association for Computing Machinery, New York, NY, USA, 545–554. https://doi.org/10.1145/3077136.3080805

[13] Jun Wang Yong Yu Lantao Yu, Weinan Zhang. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI 2017*.

[14] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *31st International Conference on Machine Learning, ICML 2014*.

[15] Shuqi Lu, Zhicheng Dou, Xu Jun, Jian-Yun Nie, and Ji-Rong Wen. 2019. PSGAN: A Minimax Game for Personalized Search with Limited and Noisy Click Data. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 555–564. https://doi.org/10.1145/3331184.3331218

[16] Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, 1412–1421. https://doi.org/10.18653/v1/D15-1166

[17] Rodrygo L.T. Santos. 2012. Explicit Web Search Result Diversification. *SIGIR Forum* 47, 1 (June 2012), 67–68. https://doi.org/10.1145/2492189.2492205

[18] 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 '10)*. Association for Computing Machinery, New York, NY, USA, 881–890. https://doi.org/10.1145/1772690.1772780

[19] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. Association for Computing Machinery, New York, NY, USA, 515–524. https://doi.org/10.1145/3077136.3080786

[20] Xiaojie Wang, Zhicheng Dou, Tetsuya Sakai, and Ji-Rong Wen. 2016. Evaluating Search Result Diversity Using Intent Hierarchies. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16)*. Association for Computing Machinery, New York, NY, USA, 415–424. https://doi.org/10.1145/2911451.2911497

[21] Xiaojie Wang, Ji-Rong Wen, Zhicheng Dou, Tetsuya Sakai, and Rui Zhang. 2017. Search Result Diversity Evaluation Based on Intent Hierarchies. *IEEE Transactions on Knowledge and Data Engineering* PP (07 2017), 1–1. https://doi.org/10.1109/TKDE.2017.2729559

[22] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise Approach to Learning to Rank: Theory and Algorithm. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*. Association for Computing Machinery, New York, NY, USA, 1192–1199. https://doi.org/10.1145/1390156.1390306

[23] 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 (SIGIR '15)*. Association for Computing Machinery, New York, NY, USA, 113–122. https://doi.org/10.1145/2766462.2767710

[24] 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 '16)*. Association for Computing Machinery, New York, NY, USA, 395–404. https://doi.org/10.1145/2911451.2911498

[25] Yisong Yue and Thorsten Joachims. 2008. Predicting Diverse Subsets Using Structural SVMs. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*. Association for Computing Machinery, New York, NY, USA, 1224–1231. https://doi.org/10.1145/1390156.1390310

[26] Yadong Zhu, Yanyan Lan, Jiafeng Guo, Xueqi Cheng, and Shuzi Niu. 2014. Learning for Search Result Diversification. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '14)*. Association for Computing Machinery, New York, NY, USA, 293–302. https://doi.org/10.1145/2600428.2609634