IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models

Anonymous Authors

ABSTRACT

This paper provides a unified account of two schools of thinking in information retrieval modelling: the generative retrieval focusing on predicting relevant documents given a query, while discriminative retrieval focusing on predicting relevancy given a query-document pair. We propose a game theoretical *minimax game* to iteratively optimise both models. On one hand, the discriminative model, aiming to mine signals from labelled and unlabelled data, provides guidance to train the generative model towards fitting the underlying relevance distribution over documents given the query. On the other hand, the generative model, acting as an attacker to the current discriminative model, generates difficult examples for the discriminative model in an adversarial way by minimising its discrimination objective. With the competition between these two models, we show that the unified framework takes advantage of both schools of thinking: (i) the generative model learns to fit the relevance distribution over documents via the signal from the discriminative model, and (ii) the discriminative model is able to exploit the unlabelled data selected by the generative model to achieve a better estimation for document ranking. Our experimental results have demonstrated significant improvement gains as much as 23.96% on Precision@5 and 15.50% on MAP over strong baselines in a variety of applications including web search, item recommendation, and question answering.

KEYWORDS

Adversarial Training, Information Retrieval, Web Search, Recommender Systems, Question Answering

ACM Reference format:
Anonymous Authors. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. In *Proceedings of ACM conference, Tokyo, Japan, August 7–11, 2017 (SIGIR '17),* 11 pages.
DOI: 10.475/123_4

1 INTRODUCTION

A typical formulation of information retrieval (IR) is to provide a (rank) list of documents given a query. It has a wide range of applications from text retrieval [1] and web search [3, 19] to recommender systems [21, 35], question answering [9], and personalised ads [27], to name just a few.

There are, arguably, two major schools of thinking when coming to IR theory and modelling [1]. The classic school of thinking is to assume that there is an underlying stochastic **generative** process between documents and information needs (clued by a query) [22]. In text retrieval, the classic relevance model of information retrieval is focused on answering how a (relevant) document is generated from a given information need: $q \rightarrow d$, where q is a query (e.g., keywords, user profiles, questions, depending on specific IR applications) and d is its corresponding document (e.g., a textual document, an information item, an answer) and the arrow indicates the direction of generation. Notable examples include Robertson and Sparck Jones's Binary Independence Model, of which each word token is independently generated to form a relevant document [36]. Language models of text retrieval consider a reverse generative process from a document to a query: $d \rightarrow q$, typically generating query terms from a document (i.e, the query likelihood function) [32, 47]. In related work of word embedding, word tokens are generated from their context words [28]. In the application of recommender systems, we also see a recommended target item (in the original document identifier space) can be generated/selected from known context items [2].

The modern school of thinking in IR recognises the strength of machine learning and shifts to a **discriminative** (classification) solution learned from labelled relevant judgements or their proxy such as clicks or ratings. It considers documents and queries jointly as features and predicts their relevancy or rank order labels from a large amount of training data: $q+d \rightarrow r$, where r denotes relevance and symbol + denotes the combining of features. A significant development in web search is learning to rank (LTR) [3, 19], a family of machine learning problems where the training objective is to provide the right ranking order of a list of documents (or items) for a given query (or context) [24]. Three major paradigms for learning to rank techniques are pointwise, pairwise and listwise methods. Pointwise methods learn to approximate the relevance estimation of each document to the human rating [23, 31]. Pairwise methods aim to distinguish the more-relevant document from a document pair [3]. Listwise methods learn to optimise the (smoothed) loss function defined over the whole ranking list for each query [4, 6]. Similarly, a recent advance in recommender systems is matrix factorisation where learned user features and item features are dot-producted together to make the prediction of the relevancy [21, 35, 45].

While the generative models of information retrieval are theoretically sound and are very successful in modelling features (e.g., text statistics, distribution over document identifier space), they suffer from the difficulty of leveraging relevancy signals from other channels such as links, clicks etc., which are largely observable in Internet-based applications. While the discriminative models of IR such as learning to rank are able to learn a retrieval ranking function implicitly from a large amount of labelled/unlabelled data, they lack a proper way of generating useful features or gathering helpful signals from large amount of unlabelled data, in particular, from text statistics derived from both documents and queries, or the distribution of the relevant documents from the data.

In this paper, we consider the generative and discriminative retrieval models as two sides of the same coin. Inspired by Generative Adversarial Nets (GANs) in machine learning [13], we propose a

SIGIR '17, Tokyo, Japan

DOI: 10.475/123_4

game theoretical *minimax game* to combine the two schools of thinking. Specifically, we define a common retrieval function (e.g., discrimination-based objective function) for both models. On one hand, the discriminative model $p_\phi(r|q,d)$ aims to maximise the objective function by learning from labelled data. It naturally provides alternative guidance to the generative retrieval model beyond traditional log-likelihood. On the other hand, the generative retrieval model $p_\theta(d|q,r)$ acts as a challenger who constantly pushses the discriminator to its limit. Iteratively it provides the most difficult cases for the discriminator to retrain by adversarially minimising the objective function. In such a way, the two types of IR models act as two players in a minimax game and each of them strikes to improve itself to 'beat' the other one at every round of this competition. Note that our minimax game based approach is fundamentally different from the existing game-theoretic IR methods [26, 46], in the sense that the existing approaches generally try to model the interaction between user and system, whereas our approach aims to unify generative and discriminative IR models.

Empirically, we have realised the proposed minimax retrieval framework in three typical IR applications: web search, item recommendation, and question answering. In our experiments, we found that the minimax game arrives at different equilibria and thus different effect of unification in different settings. With the pointwise adversarial training, the generative retrieval model can be significantly boosted by the training rewards from the discriminative retrieval model. The resulting model outperforms several strong baselines as much as 22.56% in web search and 14.38% in item recommendation on Precesion@5. We also found that with new pairwise adversarial training, the discriminative retrieval model is largely boosted by examples selected by the generative retrieval model and outperforms the compared strong algorithms up to 23.96% on Precision@5 in web search and 3.23% on Precision@1 in question answering.

2 IRGAN FORMULATION

In this section, we take the inspiration from GANs and build a unified framework of fusing generative and discriminative IR in an adversarial setting; we call it IRGAN, and its applications on specific IR problems shall be given in the follow-up section.

2.1 A Minimax Retrieval Framework

Without loss of generality, let us consider the following information retrieval problem. We have a set of queries $\{q_1, ..., q_N\}$ and a set of documents $\{d_1, ..., d_M\}$. In a general setting, a query is any specific form of the user's information need such as search keywords, a user profile, or a question, while documents could be textual documents, information items, or answers, depending on the specific retrieval task. For a given query q_n, we have a set of relevant documents labelled, denoted as set R_n. The size of R_n is much smaller than the total number of documents M.

The underlying *true* relevance distribution can be expressed as conditional probability: $p_{\text{true}}(d|q,r)$, which depicts the (user's) relevance preference distribution over the candidate documents with respect to her submitted query. Given a set of samples from $p_{\text{true}}(d|q,r)$ observed as the training data, we can try to construct two types of IR models:

Generative retrieval model $p_\theta(d|q,r)$, which tries to generate (or say select) relevant documents, from the candidate pool for the given query q; in other words, its goal is to approximate the true relevance distribution over documents $p_{\text{true}}(d|q,r)$ as much as possible.

Discriminative retrieval model $f_\phi(q,d)$, which, in contrary, tries to discriminate well-matched query-document tuples (q,d) from ill-matched ones, where the goodness of matching given by $f_\phi(q,d)$ depends on the relevance of d to q; in other words, its goal is to distinguish between relevant documents and non-relevant documents for the query q as accurately as possible. It is in fact simply a binary classifier, and we could use 1 as the class label for the query-document tuples that truly match (positive examples) while 0 as the class label for those do not really match (negative examples).

2.1.1 Overall Objective. Thus, inspired by the idea of GAN, we aim to unify these two different types of IR models by letting them play a minimax game: the generative retrieval model would try to generate (or select) relevant documents that look like the ground-truth relevant documents and therefore could fool the discriminative retrieval model, whereas the discriminative retrieval model would try to draw a clear distinction between the ground-truth relevant documents and the generated ones made by its opponent generative retrieval model. Formally, we have:

$$J^{G^*,D^*} = \min_\theta \max_\phi \sum_{n=1}^{N} \Big(\mathbb{E}_{d\sim p_{\text{true}}(d|q_n,r)} [\log D(d|q_n)] + \mathbb{E}_{d\sim p_\theta(d|q_n,r)} [\log(1 - D(d|q_n))] \Big), \quad (1)$$

where the generator G is directly written as $p_\theta(d|q_n,r)$, and the probability of document d being relevant to query q is given by the sigmoid function of the discriminator score

$$D(d|q) = \sigma(f_\phi(d,q)) = \frac{\exp(f_\phi(d,q))}{1+\exp(f_\phi(d,q))}. \quad (2)$$

We leave the specific parametrisation of $f_\phi(d,q)$ for the next section when we discuss each of three specific IR tasks we have considered. From Eq. (1), we can see that the optimal parameters of the generative retrieval model and the discriminative retrieval model can be learned iteratively by maximising and minimising the same objective function, respectively.

2.1.2 Optimising Discriminative Retrieval. The objective for the discriminator is to *maximise* the log-likelihood of correctly distinguishing the true and generated relevant documents. With the observed relevant documents, and the ones sampled from the current optimal generative model $p_{\theta^*}(d|q,r)$, one can then obtain the optimal parameters for the discriminative retrieval model:

$$\phi^* = \arg\max_\phi \sum_{n=1}^{N} \Big(\mathbb{E}_{d\sim p_{\text{true}}(d|q_n,r)} \left[\log(\sigma(f_\phi(d,q_n))\right] + \mathbb{E}_{d\sim p_{\theta^*}(d|q_n,r)} \left[\log(1-\sigma(f_\phi(d,q_n)))\right] \Big), \quad (3)$$

where if the function f_ϕ is differentiable with respect to ϕ, the above is solved typically by stochastic gradient descent.

2.1.3 Optimising Generative Retrieval. By contrast, the generative retrieval model $p_\theta(d|q,r)$ intends to *minimise* the objective; it fits the underlying relevance distribution over documents $p_{\text{true}}(d|q,r)$ and based on it, randomly samples documents from the whole document set in order to *fool* the discriminative retrieval model.

It is worth mentioning that unlike GAN [13, 18], we design the generative model to directly generate known document (in the document identifier space) not its features, because our work here intends to select relevant documents from a given document pool. Note that it is feasible to generate new documents (features, such as the value of BM25) by IRGAN, but to stay focused, we leave it for future investigation.

Specifically, while keeping the discriminator $f_\phi(q,d)$ fixed after its maximisation operation in Eq. (1), we then learn the generative model via performing its minimisation:

$$\begin{aligned}\theta^* &= \arg\min_\theta \sum_{n=1}^{N} \Big(\mathbb{E}_{d\sim p_{\text{true}}(d|q_n,r)} \left[\log \sigma(f_\phi(d,q_n))\right] + \\ &\qquad \mathbb{E}_{d\sim p_\theta(d|q_n,r)} \left[\log(1-\sigma(f_\phi(d,q_n)))\right] \Big) \\ &= \arg\max_\theta \sum_{n=1}^{N} \underbrace{\mathbb{E}_{d\sim p_\theta(d|q_n,r)} \left[\log(1+\exp(f_\phi(d,q_n)))\right]}_{\text{denoted as } J^G(q_n)}, \quad (4)\end{aligned}$$

where for each query q_n we denote the objective function of the generator as $J^G(q_n)$[1].

As the sampling of d is discrete, it cannot be directly optimised by gradient descent from the original GAN formulation. A common approach is to use policy gradient based reinforcement learning (REINFORCE) [42, 44]. Its gradient is derived as follows:

$$\begin{aligned}&\nabla_\theta J^G(q_n) \\ &= \nabla_\theta \mathbb{E}_{d\sim p_\theta(d|q_n,r)} \left[\log(1+\exp(f_\phi(d,q_n)))\right] \\ &= \sum_{i=1}^{M} \nabla_\theta p_\theta(d_i|q_n,r) \log(1+\exp(f_\phi(d_i,q_n))) \\ &= \sum_{i=1}^{M} p_\theta(d_i|q_n,r) \nabla_\theta \log p_\theta(d_i|q_n,r) \log(1+\exp(f_\phi(d_i,q_n))) \\ &= \mathbb{E}_{d\sim p_\theta(d|q_n,r)} \left[\nabla_\theta \log p_\theta(d|q_n,r) \log(1+\exp(f_\phi(d,q_n)))\right] \\ &\simeq \frac{1}{K} \sum_{k=1}^{K} \nabla_\theta \log p_\theta(d_k|q_n,r) \log(1+\exp(f_\phi(d_k,q_n))), \quad (5)\end{aligned}$$

where we perform a sampling approximation in the last step in which d_k is the k-th document sampled from the current version of generator $p_\theta(d|q_n,r)$. With reinforcement learning terminology, the term $\log(1+\exp(f_\phi(d,q_n)))$ acts as the reward for the policy $p_\theta(d|q_n,r)$ taking an action d in the environment q_n [39].

In order to reduce variance during the REINFORCE learning, we also replace the reward term $\log(1+\exp(f_\phi(d,q_n)))$ by its advantage function:

$$\log(1+\exp(f_\phi(d,q_n))) - \mathbb{E}_{d\sim p_\theta(d|q_n,r)} \left[\log(1+\exp(f_\phi(d,q_n)))\right],$$

where the term $\mathbb{E}_{d\sim p_\theta(d|q_n,r)} \left[\log(1+\exp(f_\phi(d,q_n)))\right]$ acts as the baseline function in policy gradient [39].

The overall logic of our proposed IRGAN solution is summarised in Algorithm 1. Before the adversarial training, the generator and discriminator can be initialised by their conventional models. Then during the adversarial training stage, the generator and discriminator are trained alternatively via Eqs. (5) and (3).

Algorithm 1 Minimax Game for IR (a.k.a IRGAN)

Input: generator $p_\theta(d|q,r)$; discriminator $f_\phi(\boldsymbol{x}_i^q)$;
training dataset $\mathcal{S} = \{\boldsymbol{x}\}$

```
Initialise p_θ(d|q, r), f_φ(q, d) with random weights θ, φ.
Pre-train p_θ(d|q, r), f_φ(q, d) using S
repeat
    for g-steps do
        p_θ(d|q, r) generates K documents for each query q
        Update generator parameters via policy gradient Eq. (5)
    end for
    for d-steps do
        Use current p_θ(d|q, r) to generate negative examples and combine with given positive examples S
        Train discriminator f_φ(q, d) by Eq. (3)
    end for
until IRGAN converges
```

2.2 Extension to Pairwise Case

In many IR problems, it is common that the labelled training data available for learning to rank are not a set of relevant documents but a set of ordered document pairs for each query, as it is often easier to capture users' relative preference judgements on a pair of documents than their absolute relevance judgements on individual documents (e.g., from a search engine's click-through log) [19]. Furthermore, if we use graded relevance scales (indicating a varying degree of match between each document and the corresponding query) rather than binary relevance, the training data could also be represented naturally as ordered document pairs.

Here we show that our proposed IRGAN framework would also work in such a pairwise setting for learning to rank, where for each query q_n we have a set of labelled document pairs $R_n = \{\langle d_i, d_j\rangle | d_i \succ d_j\}$, where $d_i \succ d_j$ means d_i is more relevant to q_n than d_j. As we have done in Section 2.1, let $p_\theta(d|q,r)$ and $f_\phi(q,d)$ denote the generative retrieval model and the discriminative retrieval model respectively.

The generator G would try to generate document pairs that are similar to those in R_n, i.e., with the correct ranking. The discriminator D would try to distinguish such generated document pairs from those real document pairs. The probability that a document pair $\langle d_u, d_v\rangle$ being correctly ranked can be estimated by the discriminative retrieval model through a sigmoid function:

$$\begin{aligned}D(\langle d_u, d_v\rangle|q) &= \sigma(f_\phi(d_u,q) - f_\phi(d_v,q)) \\ &= \frac{\exp(f_\phi(d_u,q) - f_\phi(d_v,q))}{1+\exp(f_\phi(d_u,q) - f_\phi(d_v,q))} \\ &= \frac{1}{1+\exp(-z)}, \quad (6)\end{aligned}$$

where $z = f_\phi(d_u,q) - f_\phi(d_v,q)$. Note that $-\log D(\langle d_u, d_v\rangle|q) = \log(1+\exp(-z))$ is exactly the pairwise ranking loss function used

[1] Following [13], $\mathbb{E}_{d\sim p_\theta(d|q_n,r)}[\log(\sigma(f_\phi(d,q_n)))]$ is normally used to be maximised instead, which keeps the same fixed point but provides more sufficient gradient for the generative model.

by the learning to rank algorithm RankNet [3]. In addition to the logistic function $\log(1+\exp(-z))$, it is possible to make use of other pairwise ranking loss functions [7], such as the hinge function $(1-z)_+$ (as used in Ranking SVM [16]) and the exponential function $\exp(-z)$ (as used in RankBoost [11]), to define the probability $D(\langle d_u, d_v\rangle|q)$.

If we use the standard cross entropy cost for this binary classifier as before, we have the following minimax game:

$$J^{G^*,D^*} = \min_\theta \max_\phi \sum_{n=1}^{N} \Big(\mathbb{E}_{\mathbf{o}\sim p_{\text{true}}(\mathbf{o}|q_n)} [\log D(\mathbf{o}|q_n)] + \mathbb{E}_{\mathbf{o}'\sim p_\theta(\mathbf{o}'|q_n)} [\log(1 - D(\mathbf{o}'|q_n))] \Big), \tag{7}$$

where $\mathbf{o} = \langle d_u, d_v\rangle$ and $\mathbf{o}' = \langle d'_u, d'_v\rangle$ are true and generated document pairs for query q_n respectively.

In practice, to generate a document pair through generator G, we first pick a document pair $\langle d_i, d_j\rangle$ from R_n, take the lower ranked document d_j, and then pair it with a document d_k selected from the unlabelled data to make a new document pair $\langle d_k, d_j\rangle$. The underlying rationale is that we are more interested in identifying the documents similar to higher ranked document d_i as such documents are more likely to be relevant to the query q_n. The selection of the document d_k is based on the criterion that d_k should be more relevant than d_j according to the current generative model $p_\theta(d|q,r)$. In other words, we would like to select d_k from the whole document set to generate a document pair $\langle d_k, d_j\rangle$ which can imitate the document pair $\langle d_i, d_j\rangle \in R_n$.

Suppose that the generative model $p_\theta(d|q,r)$ is given by a softmax function (which is indeed used throughout Section 3, as we shall see later)

$$p_\theta(d_k|q,r) = \frac{\exp(g_\theta(q,d_k))}{\sum_d \exp(g_\theta(q,d))}, \tag{8}$$

where $g_\theta(q,d)$ is a task-specific real-valued function reflecting the chance of d being generated from q. The probability of choosing a particular document d_k could then be given by another softmax function:

$$\begin{aligned} G(\langle d_k, d_j\rangle|q) = p_\theta(\mathbf{o}'|q) &= \frac{\exp\left(g_\theta(d_k,q) - g_\theta(d_j,q)\right)}{\sum_d \exp\left(g_\theta(d,q) - g_\theta(d_j,q)\right)} \\ &= \frac{\exp\left(g_\theta(d_k,q)\right)}{\sum_d \exp\left(g_\theta(d,q)\right)} \\ &= p_\theta(d_k|q,r)\ . \end{aligned} \tag{9}$$

In this special case, $G(\langle d_k, d_j\rangle|q)$ happens to be equal to $p_\theta(d_k|q,r)$, which is simple and reasonable. In general, the calculation of $G(\langle d_k, d_j\rangle|q)$ probably involves both $p_\theta(d_k|q,r)$ and $p_\theta(d_j|q,r)$. For example, one alternative way is to sample d_k only from the documents more relevant to the query than d_j, and let $G(\langle d_k, d_j\rangle|q)$ be directly proportional to $\max(p_\theta(d_k|q,r) - p_\theta(d_j|q,r), 0)$.

This generative model $p_\theta(d|q,r)$ could be trained by the REINFORCE algorithm [42, 44] in the same fashion as we have explained in Section 2.1.

2.3 Discussion

It can be proved that when we know the true relevance distribution exactly, the above minimax game of IRGAN, both pointwise and pairwise, has a Nash equilibrium in which the generator perfectly fits the distribution of true relevant documents (i.e., $p_\theta(d|q,r) = p_{\text{true}}(d|q,r)$ in the pointwise case and $p_\theta(\mathbf{o}'|q) = p_{\text{true}}(\mathbf{o}|q)$ in the pairwise case), while the discriminator cannot distinguish generated relevant documents from the true ones (i.e., the probability of d being relevant to q, $D(d|q)$ in the pointwise case or $D(\mathbf{o}'|q)$ in pairwise, is always $\frac{1}{2}$) [13]. However, in practice, the true distribution of relevant documents is unknown, and in such a situation, how the generative/discriminative retrieval models could converge to achieve such an equilibrium is still an open problem in the current research literature [13, 14]. In our empirical study in IR, we found that depending on the specific tasks, generative retrieval and discriminative retrieval may reach a different level of performance; and at least one of them would be significantly improved compared to the one without adversarial training.

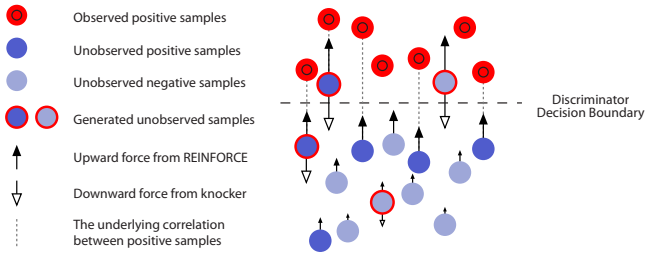

Figure 1: An illustration of IRGAN training.

In order to understand how the discriminator and the generator help each other, suppose for (part of) positive documents, their relevance score by the discriminator $f_\phi(q,d)$ and the conditional probabilistic density $p_\theta(d|q,r)$ are positively correlated. In each epoch, the generator tries to generate samples closed to the discriminator's decision boundary to confuse its training next round, while the discriminator tries to score down the generated samples. With the above assumption, since there exists positive correlation patterns between the positive but unobserved (i.e., the true-positive) samples and (part of) the observed positive samples, the generator will learn to push upwards these positive but unobserved samples again faster than other samples with the signal from the discriminator.

To understand this process further, let us take an analogy with a knocker kicking the floating soap in the water, as illustrated in Figure 1. There exists some linking lines (i.e. positive correlation) between the positive observed soaps that permanently float on the surface of the water (i.e. decision boundary of the discriminator) and the unobserved soaps which are indeed positive. The discriminator acts as the knocker that kicks down the floating-up soaps, while the generator acts as the water that *selectively* floats the soaps up to the water surface. Even if the generator cannot perfectly fit the conditional data distribution, the could be still a dynamic equilibrium, which is obtained when the distribution of the positive and negative unobserved soaps get stable over different depth of the water. Since there exists linking lines between the observed positive soaps and the unobserved positive ones, the latter ones distributed overall higher than the negative unobserved ones.

2.4 Links to Existing Work

We continue our related work discussion from Section 1 and now focus on comparing with existing work in a greater scope.

2.4.1 Generative Adversarial Nets. Generative Adversarial Nets [13] were originally proposed to generate continuous data such as images. Our work is different in the following three aspects. First, the generative retrieval process is a stochastic sampling over discrete data, i.e., the candidate documents, which is different from the deterministic generation based on the sampled noise signal in the original GAN. Specifically, as shown in Eq. (4), for each query q_n, the objective of the generative retrieval model is to minimise the expectation of the reward signal from the discriminative retrieval over the generated document distribution, while in original GAN, the reward signal is solely dependent on a single generated instance. Second, our learning process of the generative retrieval model is based on REINFORCE algorithm from the stochastic policy gradient in reinforcement learning field [44]. In IRGAN, the generative retrieval model can be regarded as an actor which takes an action of selecting a candidate document in a given environment of the query; the discriminative retrieval model can be regarded as a critic which performs a judgement whether the query-document pair is relevant enough. Third, during training, the conflicting of ground-truth documents and the generated documents is quite common because the documents are discrete and the candidate set size is finite, which departs from the continuous (infinite) space for image or the extremely huge discrete (nearly infinite) space for text sequences [44]. Fourth, we also propose a pairwise discriminative objective, which is unique for many IR problems.

Our work is also related to conditional GAN [29] where our generative and discriminative models are conditional on the given query.

2.4.2 MLE based Retrieval Models. For unsupervised learning problem that estimates the data p.d.f. $p(\boldsymbol{x})$ and supervised learning problem that estimates the conditional p.d.f. $p(y|\boldsymbol{x})$, maximising likelihood estimation (MLE) plays as a standard learning solution [30]. In IR, MLE is also widely used as an estimation method for many relevance features or retrieval models [1], such as Term Frequency (TF), Mixture Model (MM) [48] and Probabilistic Latent Semantic Indexing (PLSI) [17]. In this paper, we provide an alternative way of training and fusing retrieval models. First, the generative process is designed to fit the underlying true conditional distribution $p_{\text{true}}(d|q,r)$ with the Jensen-Shannon divergence (as demonstrated in [13]). Thus, it is natural to leverage GAN to distil a generative retrieval model to fit such an unknown conditional data distribution using the observed user feedback data. Second, the unified training scheme of two schools of IR models offers a potential of obtaining better retrieval models because (i) the generative retrieval adaptively provides different negative samples to the discriminative retrieval training, which is strategically diverse compared with the static negative sampling [3, 35] or dynamic negative sampling using the discriminative retrieval model itself [4, 49]; and (ii) the reward signal from the discriminative retrieval model provides strategic guidance of training the generative retrieval model, which is otherwise unavailable in traditional generative retrieval model training. From the generative retrieval's perspective, IRGAN is superior to traditional maximum likelihood estimation [18]. From the discriminative retrieval's perspective, IRGAN is able to exploit the use of the unlabelled data to achieve the effect of semi-supervised learning [37]. The advantages of employing two models working together get more attention in recent research; one of the variations is the *dual learning* proposed for two-agent co-learning in machine translation [43].

2.4.3 Noise-Contrastive Estimation. Our work is also related to noise-contrastive estimation (NCE) that aims to correctly distinguish the true data $(y, \boldsymbol{x}) \sim p_{\text{data}}(y|\boldsymbol{x})$ from a known noise samples $(y_n, \boldsymbol{x}) \sim p_{\text{noise}}(y_n|\boldsymbol{x})$. NCE is proved to be equivalent with MLE when the noise sample is abundant [15]. With the finite noise samples for contrastive learning, NCE is usually leveraged as an efficient learning method approximated to MLE when the latter is inefficient, for example when the p.d.f is built by large-scale softmax modelling.

Furthermore, self-contrastive estimation (SCE) [14], a special case of NCE when the noise is directly sample from the current (or a very recent) version of the model. It is proved that the gradient of SCE matches the one of MLE, with no prerequisite of infinite noise samples, which is a very promising property of SCE learning. Dynamic negative item sampling work [34, 45, 49] in top-N item recommendation with implicit feedback turns to be a practical use case of SCE, with specific solution of efficient sampling strategies.

The emergence of generative adversarial nets (GANs) [13], thus our IRGAN, opens a door of learning generative and discriminative retrieval models in an adversarial setting. Compared to NCE and SCE, the GAN paradigm enables two models to learn in an adversarial fashion, i.e. the discriminator learns to distinguish the true data from the generated (faked) one while the generator learns to generate high-quality instances to fool the discriminator.

3 APPLICATIONS

In this section, we apply our IRGAN framework into three specific IR scenarios: (i) web search with learning to rank, (ii) item recommendation, and (iii) question answering.

As formulated in Section 2, the generator's conditional distribution $p_\theta(d_i|q,r) = \exp(g_\theta(q,d_i))/\sum_{d_j}\exp(g_\theta(q,d_j))$, i.e., Eq. (8), fully depends on the scoring function $g_\theta(q,d)$. In the sampling stage, the temperature parameter τ is incorporated in Eq. (8) as

$$p_\theta(d|q,r) = \frac{\exp(g_\theta(q,d)/\tau)}{\sum_{j\in\mathcal{I}}\exp(g_\theta(q,d)/\tau)}, \tag{10}$$

where the lower temperature leads to the sampling focusing more on top-ranked documents. A special case is when the temperature is set as 0, which means the entropy of the generator is 0. In such a case the generator simply ranks the documents in descendant order and selects the top ones.

The discriminator's estimation and ranking of the documents, i.e., Eq. (2) for pointwise setting and Eq. (6) for pairwise setting, depends on the scoring function $f_\phi(q,d)$.

The implementation of the two scoring functions $g_\theta(q,d)$ and $f_\phi(q,d)$ are task-specific. Although there could be various implementations of $f_\phi(q,d)$ and $g_\theta(q,d)$, e.g., $f_\phi(q,d)$ is implemented via a three-layer neural work while $g_\theta(q,d)$ is implemented via factorisation machine [33], to focus more on the adversarial training, in this section we choose to implement them using the same function (with different sets of parameters)[2]:

$$g_\theta(q,d) = s_\theta(q,d) \quad \text{and} \quad f_\phi(q,d) = s_\phi(q,d), \tag{11}$$

[2]We will, however, provide a dedicated experiment on the interplay of the two players from scoring functions with different model complexity in Section 4.1.

and in the following subsections we will discuss the implementation of the relevance scoring function $s(q, d)$ for the three studied scenarios.

3.1 Web Search

Generally speaking, there are three types of loss functions designed for web search learning to rank, namely, pointwise [31], pairwise [3] and listwise [6]. To our knowledge, the listwise approaches with a loss defined on a document pair and a list-aware weight added on the document pair perform practically effectively across various learning to rank tasks, e.g., LambdaRank [5] and LambdaMART [4]. Despite of the variety of ranking loss functions, almost every learning to rank solution is based on a scoring function $s(q, d)$.

In web search scenario, each query-document pair (q, d) can be represented by a vector $\boldsymbol{x}_{q,d} \in \mathbb{R}^k$, where each dimension represents some statistical value of the query-document pair or the either part, such as BM25, PageRank, TFIDF, language model score etc. We follow the work of RankNet [3] to implement a two-layer neural network for the score function:

$$s(q, d) = \boldsymbol{w}_2^\top \tanh(\boldsymbol{W}_1 \boldsymbol{x}_{q,d} + \boldsymbol{b}_1) + w_0 \,, \tag{12}$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{l \times k}$ is the first fully-connected layer matrix, $\boldsymbol{b}_1 \in \mathbb{R}^l$ is the bias vector for the hidden layer, $\boldsymbol{w}_2 \in \mathbb{R}^l$ and w_0 is the weights of the output layer.

3.2 Item Recommendation

Item recommendation is a popular data mining task that can be regarded as a generalised information retrieval problem, where the query is the user profile built by their past item consumption. Collaborative filtering is one of the most important methodologies for recommender systems, which explores underlying user-user or item-item similarity and based on which performs personalised recommendations [41]. In collaborative filtering, a widely adopted model is matrix factorisation [21], following which we define our scoring function for the preference of user u (i.e. the query) to item i (i.e. the document) as

$$s(u, i) = b_i + \boldsymbol{v}_u^\top \boldsymbol{v}_i, \tag{13}$$

where b_i is the bias term for item i, $\boldsymbol{v}_u, \boldsymbol{v}_i \in \mathbb{R}^k$ are the latent vectors of user u and item i respectively, defined in the k-dimensional continuous space. Here we omit the global bias and user bias as they are reduced in the task of top-N item recommendation for each user[3].

To keep our discussion uncluttered, we have chosen a basic matrix factorisation model to implement, and it is straightforward to replace it with more sophisticated models such as factorisation machines [33] and neural networks [8], whenever needed.

3.3 Question Answering

In question answering (QA) tasks [9], each question q and answer a are represented as a sequence of words. Typical QA solutions aim to understand the natural language question first and then select/generate one or more answers which best match the question [9]. Among various QA tasks, the document-based QA task is regarded as a typical ranking process based on the matching score between two pieces of texts (for question and answer, respectively) [9]. Recently, end-to-end approaches to short text pairs have been proposed, by utilising neural networks, such as convolutional neural network (CNN) [9, 38] or long short-term memory neural network (LSTM) [40].

For a question-answer pair (q, a), we define a relevance score. Specifically, one can leverage the convolutional neural networks (CNN) to learn the representation of a word sequence [20]. Each word is embedded as a vector in $\mathbb{R}^k$. By aligning the vectors of the words, an l-word sentence can be represented as a matrix in $\mathbb{R}^{l \times k}$ of embedding vectors. Then a represent vector of the current sentence is through a max-pooling-over-time strategy after convolution operation over the embedding vector aligned matrix, yielding $\boldsymbol{v}_q$ and $\boldsymbol{v}_a \in \mathbb{R}^z$, where z is the number of convolutional kernels. And the relevance score of such question-answer pair can be defined as the cosine similarity, i.e.,

$$s(q, a) = \cos(\boldsymbol{v}_q, \boldsymbol{v}_a) = \frac{\boldsymbol{v}_q^\top \boldsymbol{v}_a}{|\boldsymbol{v}_q| \cdot |\boldsymbol{v}_a|} \,. \tag{14}$$

With the sentence representation and scoring function defined above, the question answering problem is transformed into a query-document scoring problem in IR [38].

Our IRGAN solution would load pre-trained models as the initial parameters for both generator and discriminator. For each real question-answer pair, an incorrect question-answer will be sampled from the whole candidate answer set.

4 EXPERIMENTS

We have conducted our experiments[4] corresponding to the three real-world applications of our proposed IRGAN as discussed, i.e., web search, item recommendation and question answering. As each of the three applications has its own background and baselines algorithms, we organise the experiment part into three self-contained subsections. We first test both the IRGAN-pointwise and IRGAN-pairwise formulations within a single task, web search; and then IRGAN-pointwise is further investigated in the case where the rank bias is less critical (the item recommendation task), while IRGAN-pairwise is on the question answering task for which the rank bias is more critical (only answer is correct).

4.1 Web Search

4.1.1 Experiment Setup. Webpage search is an important problem in the IR field. In this part we use the well-known benchmark LETOR (LEarning TO Rank) dataset [25] for webpage ranking to conduct our experiment.

Although the standard learning to rank tasks involve explicit expert ratings for query-document pairs, implicit feedback such as click is much more common in practical applications, which means in the dataset we usually face with a (small) part of user-interacted data and a large amount of unlabelled data. Partially-labelled data provides samples which are not explicitly labelled as positive in training sets while being labelled as positive in test set. We are expecting more unobserved data without explicit positive label, which can be explored by our IRGAN methodology. Thus we make use of the MQ2008-semi collections in LETOR 4.0 as it contains both labelled and a large amount of unlabelled data.

[3] The user bias could be used as a good baseline function for the advantage function in policy gradient (Eq. (5)) to reduce the learning volatility [39].

[4] Repeatable experiment code will be published on Github upon paper acceptance.

Each query-document pair consists of a relevance level (−1, 0, 1 or 2). The higher the relevance label, the more relevant the query-document pair. Specifically, −1 means as unknown label. A query-document pair is represented by a 46-dimensional feature vector, such as BM25 and LMIR. For the purpose of using implicit feedbacks, we re-label all the query-document pairs with relevance level higher than 0 as 1, which represent the positive documents, and we re-label all other pairs with −1 or 0 relevance level 0 as unknown documents. Through our statistics, there are 784 unique queries in this dataset, on average each query is associated with about 5 positive documents and about 1,000 unknown documents. For training and test data splitting, we perform a 4:1 random splitting. Both pointwise and pairwise IRGANs are evaluated based on this dataset.

Similar to the RankNet model [3], we adopt a neural network model with one hidden layer and tanh activation to learn the query-document matching score, and the size of hidden layer equals to the feature size. Besides, both the generator and discriminator are trained from scratch.

For performance comparison, we compare the generative retrieval model in our IRGAN framework with simple RankNet [3], LambdaRank [5] and the strong baseline LambdaMART [4] for which we use the RankLib[5] implementation. For the evaluation of the compared algorithms, we use the standard ranking measures [7] such as Precision@N, Normalised Discounted Cumulative Gain (NDCG@N), Mean Average Precision (MAP) and Mean Reciprocal Ranking (MRR).

4.1.2 Results and Discussions. First we provide the overall performance of the compared approaches on the MQ2008-semi dataset as shown in Table 1. In our IRGAN framework, we use the generative retrieval model to predict the distribution of the user preferred documents given a query and then perform the rank, which is identical to performing the softmax sampling with temperature parameter set very close to 0. From the result we can see the performance improvements brought from our approach on all the measures.

Specifically, IRGAN-pairwise works better than IRGAN-pointwise on the metrics of Precision@3, NDCG@3 that are more focused on very top rankings, whereas IRGAN-pointwise performs better than IRGAN-pairwise on the metrics of Precision@10, NDCG@10 and MAP that give more weights to the later-ranked items. A possible explanation is that IRGAN-pointwise is targeted for, $p_{\text{true}}(d|q,r)$, the conditional distribution, which generally cares about whether a document is relevant to the query, while IRGAN-pairwise would focus on the ranking of the documents given the query.

It is worth mentioning that the data studied in our experiments is with implicit feedback, which is common in our real life applications, such as online advertising or web search. Traditional learning to rank methods like LambdaMART are not particularly effective in this type of problem, which may be due to its reliance on the ΔNDCG scoring for each document pair [5].

In addition, since the adversarial training is widely known as an effective yet unstable approach, we further investigate the learning trend of the proposed approach. Figures 2 and 3 show the typical learning curves for the generative and discriminative retrieval models in the two IRGAN frameworks respectively. Here we only show the performance of Precision@5 and NDCG@5 for discussion, and other metrics are with a similar trend. We can observe that after about 150 epoches for IRGAN-pointwise and 60 epoches for IRGAN-pairwise of the adversarial game training, both Precision@5 and NDCG@5 converge and the winner player consistently outperforms the best baseline LambdaRank.

[5]http://people.cs.umass.edu/~vdang/ranklib.html

Table 1: Webpage ranking performance comparison on MQ2008-semi dataset, where $*$ means significant improvement in a Wilcoxon signed-rank test.

	P@3	P@5	P@10	MAP
MLE	0.1556	0.1295	0.1029	0.1604
RankNet [3]	0.1619	0.1219	0.1010	0.1517
LambdaRank [5]	0.1651	0.1352	0.1076	0.1658
LambdaMART [4]	0.1368	0.1026	0.0846	0.1288
IRGAN-pointwise	0.1714	0.1657	**0.1257**	**0.1915**
IRGAN-pairwise	**0.2000**	**0.1676**	0.1248	0.1816
Impv-pointwise	3.82%	22.56%*	16.82%*	15.50%*
Impv-pairwise	21.14%*	23.96%*	15.98%	9.53%
	NDCG@3	**NDCG@5**	**NDCG@10**	**MRR**
MLE	0.1893	0.1854	0.2054	0.3194
RankNet [3]	0.1801	0.1709	0.1943	0.3062
LambdaRank [5]	0.1926	0.1920	0.2093	0.3242
LambdaMART [4]	0.1573	0.1456	0.1627	0.2696
IRGAN-pointwise	0.2065	**0.2225**	**0.2483**	**0.3508**
IRGAN-pairwise	**0.2148**	0.2154	0.2380	0.3322
Impv-pointwise	7.22%	15.89%	18.63%	8.20%
Impv-pairwise	11.53%	12.19%	13.71%	2.47%

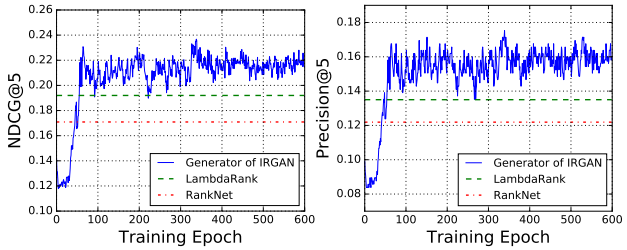

Figure 2: Learning curves of the pointwise IRGAN on web search task.

Figure 4 show how the ranking performance varies against the temperature parameter in Eq. (10) with which the generative retrieval model samples negative query-document pairs for the discriminative retrieval model. We find the empirically optimal sampling temperature is 0.2. There is an obvious performance ascent when tuning the temperature from 0 to the optimal value and an obvious descent from the optimal value to larger, which indicates *properly* increasing the aggressiveness (i.e. the degree of focusing on the top documents) of the generative retrieval model is important.

Furthermore, we study the impact of the model complexity of $f_\phi(q,d)$ and $g_\theta(q,d)$ on the interplay between them. In Figure 5 we have compared different combinations of generative and discriminative model implementations (i.e., linear model and two-layer NN) under IRGAN-pointwise and IRGAN-pairwise, respectively. We observe that (i) for IRGAN-pointwise, the NN implemented generator works better than its linear version, while the NN implemented

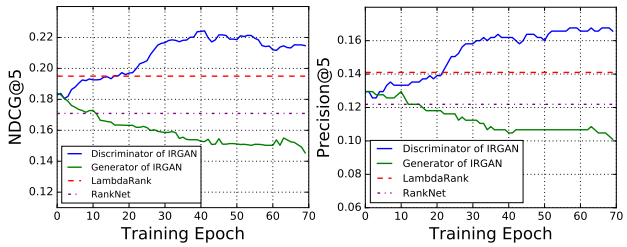

Figure 3: Learning curves of the pairwise IRGAN on web search task.

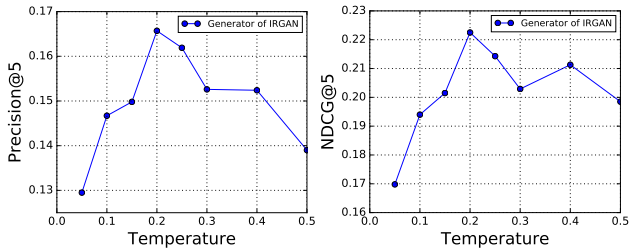

Figure 4: Ranking performance with different sampling temperatures of pointwise IRGAN on web search task.

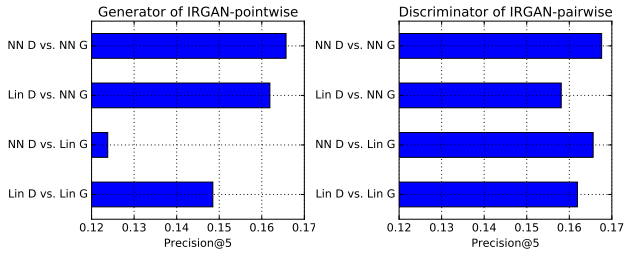

Figure 5: Ranking performance for IRGAN with different generator and discriminator scoring functions.

discriminator may not offer a good guidance if the generator has lower model complexity (i.e. linear). (ii) For IRGAN-pairwise, the NN implemented discriminator outperforms its linear version. This suggests that the one used for performing the prediction (either generator in IRGAN-pointwise or discriminator in IRGAN-pairwise) should be implemented with a capacity at least as high as its opponent.

4.2 Item Recommendation

4.2.1 Experiment Setup. We conduct our experiments on two widely used collaborative filtering datasets: Movielens (100k) and Netflix. Their details are shown in Table 2. Following the experiment setting of [49], we choose the 5-star ratings for both Movielens and Netflix as positive feedback and treat all other entries as unknown feedback as we mainly focus on the implicit feedbacks problem. For training and test data splitting, we follow [49] to give a 4:1 random splitting on both datasets. The factor numbers of matrix factorisation are 5 and 16 for Movielens and Netflix respectively.

Table 2: Characteristics of the datasets.

Dataset	Users	Items	Ratings
Movielens	943	1,683	100,000
Netflix	480,189	17,770	100,480,507

Table 3: Item recommendation results (Movielens).

	P@3	**P@5**	**P@10**	**MAP**
MLE	0.3369	0.3013	0.2559	0.2005
BPR [35]	0.3289	0.3044	0.2656	0.2009
LambdaFM [45]	0.3845	0.3474	0.2967	0.2222
IRGAN-pointwise	0.4072	0.3750	0.3140	0.2418
Impv-pointwise	5.90%*	7.94%*	5.83%*	8.82%*
	NDCG@3	**NDCG@5**	**NDCG@10**	**MRR**
MLE	0.3461	0.3236	0.3017	0.5264
BPR [35]	0.3410	0.3245	0.3076	0.5290
LambdaFM [45]	0.3986	0.3749	0.3518	0.5797
IRGAN-pointwise	0.4222	0.4009	0.3723	0.6082
Impv-pointwise	5.92%*	6.94%*	5.83%*	4.92%*

Table 4: Item recommendation results (Netflix).

	P@3	**P@5**	**P@10**	**MAP**
MLE	0.2941	0.2945	0.2777	0.0957
BPR [35]	0.3040	0.2933	0.2774	0.0935
LambdaFM [45]	0.3901	0.3790	0.3489	0.1672
IRGAN-pointwise	0.4456	0.4335	0.3923	0.1720
Impv-pointwise	14.23%*	14.38%*	12.44%*	2.87%*
	NDCG@3	**NDCG@5**	**NDCG@10**	**MRR**
MLE	0.3032	0.3011	0.2878	0.5085
BPR [35]	0.3077	0.2993	0.2866	0.5040
LambdaFM [45]	0.3942	0.3854	0.3624	0.5857
IRGAN-pointwise	0.4498	0.4404	0.4097	0.6371
Impv-pointwise	14.10%*	14.27%*	13.05%*	8.78%*

Specifically, when training the discriminative retrieval model, the generative retrieval model is leveraged to sample negative items with the sample number of positive items for each user via Eq. (10) with the temperature parameter set as 0.2, which to-some-extent pushes the item sampling to the top ones. Then the training scheme of the discriminative retrieval model is via Eq. (3). On the other hand, since the training of the generative retrieval model is performed by REINFORCE as in Eq. (5), which is normally implemented by the policy gradient on the sampled K items from $p_\theta(d|q_n, r)$. In such a case, if the item size is huge (e.g., more than 10^4) compared with K, it is more practical to leverage importance sampling to force the generative retrieval model to sample (some of) positive samples $d \in R_n$. As such the positive reward can be observed from REINFORCE and the generative retrieval model can be learned properly.

For comparison, we compare IRGAN with Bayesian Personalised Ranking (BPR) [35] and a state-of-the-art LambdaRank based collaborative filtering (LambdaFM) [45] for top-N item recommendation tasks [45, 49]. Similar to web search tasks, the evaluation measures are Precision@N, NDCG@N, MAP and MRR.

4.2.2 Results and Discussion. First, the overall performance of the compared approaches on the two datasets is shown in Tables 3

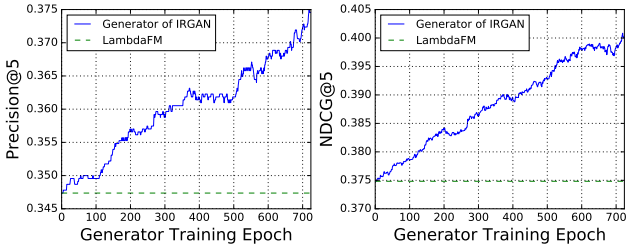

Figure 6: Learning curve of precision and NDCG of the generative retrieval model for top-5 item recommendation task on Movielens dataset.

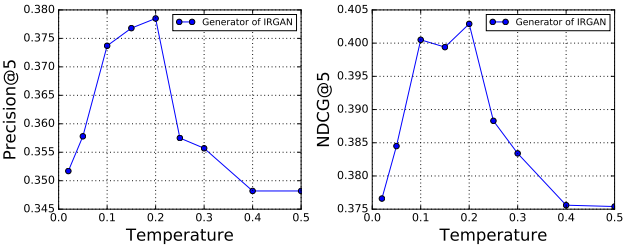

Figure 7: Ranking performance with different sampling temperatures on Movielens dataset.

and 4. From the results, we observe that IRGAN achieves statistically significant improvement across all the evaluation metrics and datasets. Note that the generative retrieval model in IRGAN does not explicitly learn to optimise the final ranking measures like what LambdaFM does, it still performs consistently better than LambdaFM. Our explanation is that the adversarial training provides both models a higher learning flexibility than the single-model LambdaFM or BPR training.

We further investigate the learning trend of the proposed approach. The learning curves are shown in Figure 6 for Precision@5 and NDCG@5. The results demonstrate a reliable training process where IRGAN owns a stable superiority over the baseline LambdaFM from the beginning of adversarial training.

In addition, as shown in Figure 7, we also investigate how the performance varies w.r.t. the sampling temperature in Eq. (10), which is consistent with the observations in web search task.

4.3 Question Answering

4.3.1 Experiment Setup. InsuranceQA [10] is one of the most studied question-answering dataset. Its questions are submitted from real users and the high-quality answers are composed by professionals with deep domain knowledge. Due to that the candidate answers are usually randomly sampled from the whole answers pool (some other QA datasets may have a small-size fixed candidate answers for each single question), InsuranceQA is suitable for testing our sampling and generating strategy. There are a training set, a development set and two test sets (test-1 and test-2) in the published corpus for a fair comparison among different methods. 12,887 questions are included in the train set with correct answers, while development set have 1,000 unseen question-answer pairs and the two test sets are composed by 1,800 pairs. The system is

Table 5: The Precision@1 of InsuranceQA.

	test-1	test-2
QA-CNN [9]	0.6133	0.5689
LambdaCNN [9, 49]	0.6183	0.5838
IRGAN-pairwise	0.6383	0.5978
Impv-pairwise	3.23%*	2.74%

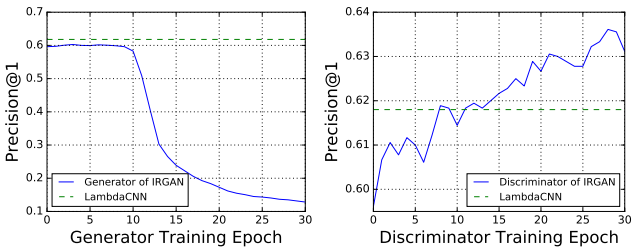

Figure 8: The experimental results in QA task.

expected to find the *only* one real answer from 500 candidate answers under the evaluation of Precision@1. As we found from the web search task that IRGAN-pairwise works better for top-ranked documents, we test IRGAN-pairwise in the QA task experiment.

To focus on evaluating the effectiveness of IRGAN, we use a simple convolutional layer on the basic embedding matrix of a question sentence or an answer sentence. A representation vector of the current sentence is distilled from a max-pooling strategy after convolution [20], yielding $\boldsymbol{v}_q$ and $\boldsymbol{v}_a$ in Eq. (14). The matching probability of such question-answer pair is denoted as the cosine distance, which is similar to the basic CNN-QA model [9].

In detail, after replacing the out-of-glossary words as a specific token, the embedding of each word is initialised as a 100-dimension random vector. In the convolutional layer, the window size of the convolution kernel is set as $(1, 2, 3, 5)$. After the convolutional layer, the max-pooling-over-time strategy is adopted [20], where each feature map will be pooled as a scalar since its convolution kernel width is the same as the embedding vector.

The pairwise adversarial training scheme for the generative and discriminative retrieval models are just the same that for web search task. The difference lies in the evaluation metric adopted. Here InsuranceQA focuses on Precision@1 of both retrieval models, since it focuses the performance of the only answer returned to the user.

4.3.2 Results and Discussion. The final results are shown as in Table 5, where IRGAN outperforms both the basic CNN model with random sampling strategy (QA-CNN) and CNN with dynamic negative sampling strategy (LambdaCNN) [9, 49]. Prior to the adversarial training process, we adopt the parameters of a strong pre-trained model to initialise the both retrieval models. The trends of the training process of two models are shown in Figure 8, which is evaluated in the test-1 set. Finally, the performance of the discriminative retrieval model of IRGAN-pairwise is better than LambdaCNN while the generative retrieval model tends to be worse during the pairwise adversarial training. The sparsity of the answers distribution may be the main reason for the worse generator, since each question only has one correct answer. Due to the sparse distribution of the true answers, the generator may fail to get a positive feedback from the discriminator.

5 CONCLUSIONS

In this paper, we have proposed IRGAN framework that unifies two schools of information retrieval methodologies, i.e., generative models and discriminative models, via adversarial training in a minimax game. Such an adversarial training framework takes advantages from both schools of methodologies: (i) the generative retrieval model is guided by the signal provided from the discriminative retrieval model, which makes it more favourable than the non-learning methods or the maximum likelihood estimation scheme; (ii) the discriminative retrieval model could be enhanced to better rank top documents via a strategic negative sampling from the generator. Overall, IRGAN provides a more flexible and strategic training environment that combines the two retrieval models. Experiments were conducted on four real-world datasets in three typical information retrieval tasks, namely web search, item recommendation and question answering. Significant performance gains were observed in each set of experiments.

Despite the great empirical success of GAN [13], there are still many questions with regard to its theoretical foundation remaining to be answered by the research community. For example, it is "not entirely clear" why GAN can generate sharper realistic images than alternative techniques [12]. Our exploration of adversarial training for IR in the proposed IRGAN framework has suggested that different equilibria could be reached in the end depending on the tasks and settings. In the pointwise version of IRGAN, the generative retrieval model gets improved more than the discriminative retrieval model, but we have an opposite observation in the pairwise case. This phenomenon certainly warrants further investigation.

We also plan to extend our framework and test it over the generation of the word tokens. One possible direction is to dive into the word weighting schemes such as [32, 36, 47] learned from the IRGAN generative retrieval model given each query and based on which derive new ranking features. Furthermore, the language model could be re-defined with GAN training, where new word patterns would be revealed.

REFERENCES

[1] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, and others. 1999. *Modern information retrieval*.

[2] Oren Barkan and Noam Koenigstein. 2016. Item2vec: neural item embedding for collaborative filtering. In *MLSP Workshop*.

[3] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to Rank Using Gradient Descent. In *ICML*.

[4] Christopher JC Burges. 2010. From RankNet to LambdaRank to LambdaMART: An Overview. *Learning* (2010).

[5] Christopher JC Burges, Robert Ragno, and Quoc Viet Le. 2006. Learning to Rank with Nonsmooth Cost Functions. In *NIPS*.

[6] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *ICML*.

[7] Wei Chen, Tie-Yan Liu, Yanyan Lan, Zhi-Ming Ma, and Hang Li. 2009. Ranking Measures and Loss Functions in Learning to Rank. In *NIPS*. 315–323.

[8] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *RecSys*.

[9] Cícero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive Pooling Networks. *CoRR* (2016).

[10] Minwei Feng, Bing Xiang, Michael R Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. In *ASRU Workshop*.

[11] Yoav Freund, Raj Iyer, Robert E Schapire, and Yoram Singer. 2003. An Efficient Boosting Algorithm for Combining Preferences. *JMLR* (2003).

[12] Ian Goodfellow. 2016. NIPS 2016 Tutorial: Generative Adversarial Networks. *arXiv preprint arXiv:1701.00160* (2016).

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *NIPS*.

[14] Ian J Goodfellow. 2014. On Distinguishability Criteria for Estimating Generative Models. *arXiv:1412.6515* (2014).

[15] Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*.

[16] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. 2000. Large Margin Rank Boundaries for Ordinal Regression. In *Advances in Large Margin Classifiers*.

[17] Thomas Hofmann. 1999. Probabilistic Latent Semantic Indexing. In *SIGIR*.

[18] Ferenc Huszár. 2015. How (not) to Train your Generative Model: Scheduled Sampling, Likelihood, Adversary? *arXiv preprint arXiv:1511.05101* (2015).

[19] Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *KDD*.

[20] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *arXiv preprint arXiv:1408.5882* (2014).

[21] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* (2009).

[22] John Lafferty and Chengxiang Zhai. 2002. Probabilistic Relevance Models Based on Document and Query Generation. In *Language Modeling and Information Retrieval*.

[23] Ping Li, Christopher JC Burges, Qiang Wu, JC Platt, D Koller, Y Singer, and S Roweis. 2007. McRank: Learning to Rank Using Multiple Classification and Gradient Boosting.. In *NIPS*.

[24] Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval* (2009).

[25] Tie-Yan Liu, Jun Xu, Tao Qin, Wenying Xiong, and Hang Li. 2007. LETOR: Benchmark Dataset for Research on Learning to Rank for Information Retrieval. In *Proceedings of SIGIR 2007 Workshop on Learning to Rank for Information Retrieval*.

[26] Jiyun Luo, Sicong Zhang, and Hui Yang. 2014. Win-win search: dual-agent stochastic game in session search. In *SIGIR*.

[27] H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, and others. 2013. Ad click prediction: a view from the trenches. In *KDD*. ACM, 1222–1230.

[28] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.

[29] Mehdi Mirza and Simon Osindero. 2014. Conditional Generative Adversarial Nets. *arXiv preprint arXiv:1411.1784* (2014).

[30] In Jae Myung. 2003. Tutorial on maximum likelihood estimation. *Journal of mathematical Psychology* (2003).

[31] Ramesh Nallapati. Discriminative Models for Information Retrieval. In *SIGIR*. ACM.

[32] Jay M Ponte and W Bruce Croft. 1998. A language modeling approach to information retrieval. In *SIGIR*.

[33] Steffen Rendle. 2010. Factorization machines. In *ICDM*.

[34] Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *WSDM*.

[35] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*.

[36] Stephen E Robertson and K Sparck Jones. 1976. Relevance weighting of search terms. *Journal of the American Society for Information science* (1976).

[37] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved Techniques for Training GANs. In *NIPS*.

[38] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *SIGIR*.

[39] Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, and others. 1999. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *NIPS*.

[40] Di Wang and Eric Nyberg. 2015. A Long Short-Term Memory Model for Answer Sentence Selection in Question Answering. In *ACL*.

[41] Jun Wang, Arjen P De Vries, and Marcel JT Reinders. 2006. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *SIGIR*.

[42] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* (1992).

[43] Yingce Xia, Di He, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual Learning for Machine Translation. In *NIPS*.

[44] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *AAAI*.

[45] Fajie Yuan, Guibing Guo, Joemon M Jose, Long Chen, Haitao Yu, and Weinan Zhang. 2016. Lambdafm: learning optimal ranking with factorization machines using lambda surrogates. In *CIKM*.

[46] ChengXiang Zhai. 2016. Towards a game-theoretic framework for text data retrieval. *IEEE Data Eng. Bull.* (2016).

[47] Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *TOIS* (2004).

[48] ChengXiang Zhai and John D. Lafferty. 2001. Model-based Feedback in the Language Modeling Approach to Information Retrieval. In *CIKM*.

[49] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *SIGIR*.