IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models [1]

Adversarial Ranking for Language Generation [2]

[1] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, Dell Zhang

[2] Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, Ming-Ting Sun

Presented by Xinyuan Zhang

March 2, 2018

Information Retrieval

A typical formulation of information retrieval (IR) is to provide a (rank) list of documents (keywords, user profiles, questions) given a query (textual documents, information items, answers).

- Generative approaches: generating documents from query terms $q \rightarrow d$ or generating query terms from documents $d \rightarrow q$.
- Discriminative approaches: taking documents and queries jointly as features to predict their relevancy or rank order labels $q + d \rightarrow r$. (Learning-to-Rank)
- IRGAN combines both two schools of thinking using Generative Adversarial Nets (GANs).

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

where the generative retrieval model G is written as $p_{\theta}(d|q_n, r)$, the probability of document d being relevant to query q which is given by the sigmoid function $D(d|q) = \sigma(f_{\phi}(q, d))$.

The generator's conditional distribution is given by

$$p_\theta(d|q,r) = \frac{\exp(g_\theta(q,d)/\tau)}{\sum_{j\in\mathcal{I}} \exp(g_\theta(q,d_j)/\tau)}, \tag{2}$$

where $g_\theta(q,d)$ is the scoring function and τ is the temperature parameter. In this case, the generator simply ranks the documents in descending order and selects the top ones.
Two functions $g_\theta(q,d)$ and $f_\phi(q,d)$ are task-specific.

- ϕ is solved by maximizing the objective function using stochastic gradient descent.
- The sampling of d is discrete so policy gradient method is adapted to solve θ. The objective function of the generator is denoted as $J^G(q_n)$.

$$\begin{aligned}
\nabla_\theta J^G(q_n) &= \nabla_\theta \mathbb{E}_{d \sim p_\theta(d|q_n,r)}[\log(1+\exp(f_\phi(d,q_n)))] && (3)\\
&\cong \frac{1}{K} \nabla_\theta \log p_\theta(d_k|q_n,r) \log(1+\exp(f_\phi(d_k,q_n))) && (4)
\end{aligned}$$

where sampling approximation is performed. With the reinforcement learning terminology, the term $\log(1+\exp(f_\phi(d_k,q_n)))$ acts as the reward for the policy $p_\theta(d|q_n,r)$ taking an action d in the state q_n.

RankGAN consists of a sequence generator G and a ranker R, where the R can endow a relative rank among the sequences when given a reference.

$$\min_{\theta} \max_{\phi} \mathbb{E}_{s \sim \mathcal{P}_h}[\log R_{\phi}(s|U, \mathcal{C}^-)] + \mathbb{E}_{s \sim G_{\theta}}[\log(1 - R_{\phi}(s|U, \mathcal{C}^+))] \quad (5)$$

where $s \sim \mathcal{P}_h$ and $s \sim G_{\theta}$ denote that s is from human-written sentences and synthesized sentences, respectively. The U is the reference set used for estimating relative ranks, and C^+, C^- are the comparison set with regard to different input sentences s.
In this paper, the generative model is designed with LSTM.

The ranking score for a certain sentence s is given by

$$P(s|u, \mathcal{C}) = \frac{\exp(\gamma\alpha(s|u))}{\sum_{s' \in \mathcal{C}'} \exp(\gamma\alpha(s'|u))} \tag{6}$$

where the relevance score function $\alpha(s|u)$ is measured by cosine similarity and γ shares the similar idea as the $1/\tau$ in quation 2. The set $\mathcal{C}' = \mathcal{C} \cup \{s\}$ denotes the set of input sentences to be ranked.
The expected log ranking score computed for the input sentence s is

$$\log R_\phi(s|U, \mathcal{C}) = \mathbb{E}_{u \in U} \log[P(s|u, \mathcal{C})] \tag{7}$$

- ϕ is solved by maximizing the objective function using stochastic gradient descent.
- The synthetic data in text generation task is based on discrete symbols, so policy gradient method is adopted to solve θ.
  The existing sequence $s_{1:t-1} = (w_0, w_1, ..., w_{t-1})$ is the current state, the next token w_t selected is an action sampling from the policy $p_\theta(w_t|s_{1:t-1})$.

$$\bigtriangledown_\theta L_\theta(s_0) = \mathbb{E}_{s_{1:T} \sim G_\theta}[\sum_{t-1}^{T} \sum_{w_t \in V} \bigtriangledown_\theta p_\theta(w_t|s_{1:t-1}) V_{\theta,\phi}(s_{1:t}, U)] \quad (8)$$

where $V_{\theta,\phi}(s_{1:t}, U)$ is the expected future reward for partial sequences and s_0 is the first generated token w_0.

Experimental Results
IRGAN

Experiments are conducted on 3 real-world applications, i.e., web search, item recommendation, and question answering.

Table 1: Webpage ranking performance comparison on the MQ2008-semi dataset, where ∗ means a significant improvement according to the 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%

Table 3: Item recommendation results (Movielens).

	P@3	P@5	P@10	MAP
MLE	0.3369	0.3013	0.2559	0.2005
BPR [34]	0.3289	0.3044	0.2656	0.2009
LambdaFM [46]	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 [34]	0.3410	0.3245	0.3076	0.5290
LambdaFM [46]	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 [34]	0.3040	0.2933	0.2774	0.0935
LambdaFM [46]	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 [34]	0.3077	0.2993	0.2866	0.5040
LambdaFM [46]	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%*

Table 5: The Precision@1 of InsuranceQA.

	test-1	test-2
QA-CNN [9]	0.6133	0.5689
LambdaCNN [9, 51]	0.6294	0.6006
IRGAN-pairwise	0.6444	0.6111
Impv-pairwise	2.38%*	1.75%

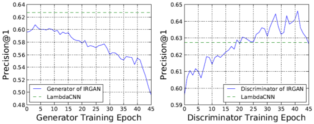

Figure 8: The experimental results in QA task.

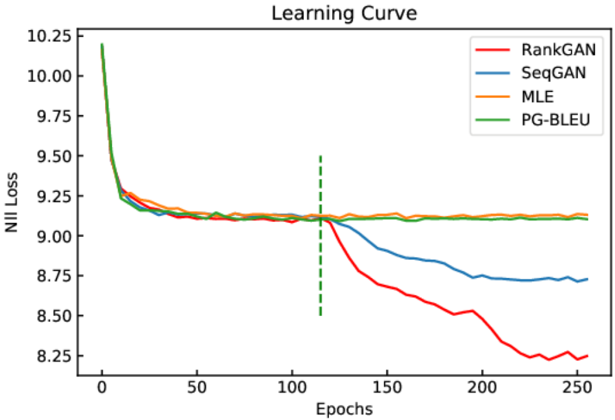

Figure: Learning curves of different methods on the simulation of synthetic data with respect to different training epochs. Note that the vertical dashed line indicates the end of the pre-training of PG-BLEU, SeqGAN and RankGAN.