UTIRL at the NTCIR-16 ULTRE Task

Anh Tran
University of Utah
Salt Lake City, Utah, USA
abtran@cs.utah.edu

Tao Yang
University of Utah
Salt Lake City, Utah, USA
taoyang@cs.utah.edu

Qingyao Ai
University of Utah
Salt Lake City, Utah, USA
aiqy@cs.utah.edu

ABSTRACT

The UTIRL team participated in both Offline and Online unbiased learning-to-rank (ULTR) (Chinese) subtasks of the NTCIR-16 ULTRE Task. This paper describes our implemented algorithms and analysis the official results. In the Offline ULTR subtask, we tried a newly proposed ULTR algorithm and an ensemble of ten models consisting of five different algorithms on two neural networks. In the Online ULTR subtask, we used three algorithms trained on a deep neural network.

KEYWORDS

unbiased learning to rank, online learning to rank, offline learning to rank

TEAM NAME

UTIRL

SUBTASKS

Online Unbiased Leaning-to-Rank (Chinese), Offline Unibiased Learning-to-Rank(Chinese

1 INTRODUCTION

Learning-to-Rank with clickthrough data has received considerable attention from both industry and academia [5]. While clickthrough data can better reflect user information need [15] and easier to collect and create large-scale training data without relying on expensive manual annotations[6], it suffered from intrinsic noise and bias as a result of user interaction such as position bias[3]. Thus, many researchers have study algorithms that could produce unbiased learning-to-rank (ULTR) models that are trained on biased users' click log.

NTCIR-16 ULTRE task[18] is a series of ULTR tasks that serves as a platform to evaluate and benchmark different ULTR approaches systematically. It also aims to examine different effects of different user simulation models on the performance of ULTR models. In the tasks, we are provided with the dataset containing query-document pairs' features, initial ranking lists as well as simulated clicks files generated from five different user simulation click models, namely, User Browsing Model (UBM), Position-Based Model (PBM), Dependent Click Model (DCM), Mobile Click Model (MCM), and Fusion. Our goal is to improve the ranking performance.

For this NTCIR-16 ULTRE task, we participated in both Offline and Online ULTR subtasks. In the Offline ULTR subtask, we implemented five different algorithms: Inverse Propensity Weight[7, 15], Dual Learning Algorithm[1], Regression EM[16], Pairwise Debiasing[4], and Propensity Ratio Scoring [14] and trained each algorithms with two different neural networks. The first network is a simple three layers deep neural network similar to the one in [2]. The second neural network, proposed by Pang et al., is called SetRank [9]. After that, we created an ensemble of these models. In addition, we also trained a separate model using Propensity Ratio Scoring algorithm as it is a recently proposed algorithm. We want to compare it with previous algorithms. We trained these models on the provided click logs generated by the different user simulation models. Based on the result, we found that Propensity Ratio Scoring models trained on UBM and MCM clicks achieved the average highest performance among all the runs in NTCIR-16 Offline ULTR subtask. In contrast, the ensemble model achieved the highest performance among all the models trained on DCM clicks. In the Online ULTR subtask, we implemented three online ULTR algorithms, namely, Dueling Bandit Gradient Descent [17], Multileave Gradient Descent[11], and Null Space Gradient Descent[13] trained on the three layers Deep Neural Network. The result showed that DBGD achieved the highest overall performance among the three algorithms.

2 OFFLINE UNBIASED LEARNING-TO-RANK SUBTASK

This section briefly discusses the well-established user examination hypothesis[10] used in different algorithm we implemented for this task, and the ranking models we implemented.

2.1 Offline Unbiased Learning-to-Rank Algorithms

The user examination hypothesis states that each document will be clicked if and only if the document is relevant to the user information need and the user has examined the document. This can be formulated as, given each document true relevance $\boldsymbol{r}$, the variable indicating whether the user has examined the documents $\boldsymbol{o}$, and the click behavior of search users $\boldsymbol{c}$, we have the following equation:

$$P(c_d = 1) = P(o_d = 1) \cdot P(r_d = 1) \tag{1}$$

For learning-to-rank task, given the scoring function f_θ and the document ground truth relevance $\boldsymbol{r}$, the local loss function is computed as follow:

$$l(f_\theta, \mathbf{r}) = \Delta(d, r_d|f_\theta) \tag{2}$$

The goal of offline ULTR is to create a loss function $l'(f_\theta, \mathbf{c})$ with the following property:

$$\mathbb{E}_o\left[l'(f_\theta, c)\right] = l(f_\theta, \mathbf{r}) \tag{3}$$

2.1.1 Inverse Propensity Weighting. Inverse Propensity Weighting (IPW) is one of the first offline ULTR algorithms [7, 15]. IPW's loss function is derived as follow:

$$l_{IPW}(f_\theta, c) = \sum_{d,c_d=1} \frac{\Delta(d, c_d|f_\theta)}{P(O_d = 1)} \tag{4}$$

$P(O_d = 1)$, which is the probability of a document being examined, is calculated by performing online result randomization, in documents in each query are randomly shuffled to ensure that each position in the rank list has equal probabilities of containing the relevant documents.

2.1.2 *Dual Learning Algorithm.*
Dual Learning Algorithm (DLA)[1] tries to conduct ULTR without using result randomization. Specifically, Ai et al.[1] suggested that in Eq. (1), the positions of o_d and r_d are interchangeable, which, theoretically, implies that the counterfactual learning of inverse propensity weighting can be simultaneously applied on both directions. Thus, DLA simultaneously trains a ranking model f_θ and an examination propensity estimation model ϕ with an inverse relevance weight loss function (IRW):

$$l_{IRW}(\phi, c) = \sum_{d, c_d=1} \frac{\Delta(d, c_d|\phi)}{P(r_d = 1)} \tag{5}$$

2.1.3 *Regression EM.*
Regression EM(REM) [16] utilizes a graphic model and EM algorithm to estimate the examination propensity and train the ranking model unifyingly. Given click log c, document click c_d and latent variables o_d and r_d, using the user examination hypothesis in Eq. (1), REM's computation for each query q's likelihood of observed clicks is:

$$\log P(c) = \sum_d c_d \log(P(o_d=1) \cdot P(r_d=1)) + (1-c_d) \log(1-P(o_d=1) \cdot P(r_d=1)) \tag{6}$$

$P(r_d = 1)$ is calculated using ranking function f_θ as:

$$P(r_d = 1) = \frac{1}{1 + \exp(-f_\theta(d))} \tag{7}$$

REM's pointwise loss function is computed as follow:

$$l(f_\theta, \boldsymbol{r}) = -\sum_d r_d \log(P(r_d = 1)) + (1 - r_d) \log(P(r_d = 1)) \tag{8}$$

2.1.4 *Pairwise Debiasing.*
The Pairwise Debiasing (PairD) model, proposed by Hu et al.[4], trains examination propensity estimation models and ranking models simultaneously. PairD computes the loss function $l(f_\theta, \mathbf{r})$ as:

$$l_{PD}(f_\theta, c) = \sum_{d^+, d^-, c_{d^+}=1, c_{d^-}=0} \frac{\Delta(f_\theta, d^+, d^-)}{P(o_d = 1) \cdot t} \tag{9}$$

where d^+ and d^- referred to click and not click documents and $r_{d^+} > r_{d^-}$. This loss is computed with the assumption that $P(c_d = 0) = t \cdot P(r_d = 1)$.

2.1.5 *Propensity Negative Scoring.*
The Propensity Negative Scoring (PRS)[14] implements a new weighting scheme that considers unclicked relevant documents to avoid relevant-relevant document comparisons in pairwise losses. The loss is computed as follow:

$$\begin{aligned} &l_{PRS}(f_\theta, \boldsymbol{c}) \\ &= \sum_{d^+, d^-, c_{d^+}=1, c_{d^-}=0} \Delta(f_\theta, d^+, d^-) \cdot \frac{P(o_{d^+} = 1)}{P(o_{d^-} = 1)} \end{aligned} \tag{10}$$

Here, d^+ and d^- referred to click and not click documents within a query. $P(o_d = 1)$ is derived using the IPW scheme.

2.2 Ranking models

This section briefly details the two ranking models used for this task.

2.2.1 *Multi-layer Perceptron.*
The Multi-layer Perception network (MLP) consists of three layers, each with 512, 256, and 128 neurons, respectively. We apply batch normalization before each layer. The MLP uses ELU as its activation function.

2.2.2 *SetRank.*
SetRank, proposed by Pang et al. [9], is multivariate ranking model inspired by Set Transformer [8]. Using a self-attention mechanism, Set Rank considers the inputted documents list as a whole instead of individual documents. As such, it can better capture the local context information between these documents and model their interrelationship in ranking. In addition, SetRank is permutation-invariant, meaning, the permutations of the inputted documents do not affect the outputted rank list. Experimental results showed that SetRank significantly outperformed traditional learning-to-rank models and state-of-the-art neural IR models. Thus, we utilize it for the ULTRE task.

We implemented our model based on the code provided in the paper using Pytorch library. The hyper-parameter settings are the same as described.

2.3 Offline ULTR Models

For the Offline subtask, we submitted two models, an ensemble model and a PRS model. These models are trained using the ULTRA toolbox's pipeline[12].

2.3.1 *Ensemble model.*
For each type of click label, we trained ten models by implementing each algorithm on the two ranking models. After that, we made an ensemble using these models. The ensemble averaged the score of all the model for each document in the query and sorted the score to generate the rank list.

2.3.2 *PRS model.*
Since PRS is a newer algorithm, we wanted to benchmark it using the ULTRE dataset. For this model, we simply trained SetRank with the PRS algorithms.

3 ONLINE UNBIASED LEARNING-TO-RANK SUBTASK

In this section, we briefly described the algorithms used for Online ULTR subtask and the ranking model used.

3.1 Online Unbiased Learning-to-Rank Algorithms

Unlike their Offline counterpart, Online ULTR algorithms we used focused on dynamically controlling the displayed rank lists for each query session to collect unbiased user feedback.

3.1.1 *Dueling Bandit Gradient Descent (DBGD).*
Proposed by Yue and Joachims [17], optimized the ranking function f_θ by adding perturbation parameter θ', usually sampled uniformly, to parameter θ to create a new ranking function f'_θ in each training step. The rank lists produced by these two functions are then shown to real user (directly or interleavedly). Clicks collected are used to calculate the losses, and based on these losses, the model is updated accordingly.

Table 1: ULTRE task result with the overall performance ranking for each User Simulation Models

		PBM		DCM		UBM		MCM		FUSION		AVG	
Offline ULTRE subtask result	PRS	0.7905	4	0.7930	5	**0.8026**	1	**0.7889**	1	**0.7947**	3	**0.7939**	1
	Ensemble	**0.7913**	3	**0.8147**	1	0.7913	2	0.7784	5	0.7827	6	0.7917	3
Online ULTRE subtask result	DBGD	0.7545	4	0.7768	4	0.7606	5	0.7863	2	0.7395	6	0.7635	5
	MGD	0.7431	5	0.7087	6	0.7146	7	0.7596	5	0.7563	6	0.7365	6
	NSGD	0.6858	7	0.6938	7	0.7393	6	0.7371	7	0.7696	5	0.7251	7

3.1.2 Multileave Gradient Descent. Multileave Gradient Descent (MGD) [11] is an extension of DBGD. The main difference is that MGD sampled multiple perturbation parameters θ' to find a better candidates selection.

3.1.3 Null Space Gradient Descent. Null Space Gradient Descent [13] is another extension of DBGD that also implements multiple perturbation parameters θ'. NSGD stores previous training instances perturbed parameters that resulted in poorly performing gradient and samples new ones from null space for more efficient direction exploration.

3.2 Online ULTR Models

For each click type, we trained the three online algorithms with SetRank. Similar to the Offline subtask, we also utilized the ULTRA toolbox [12] for training the models. However, instead of using ULTRA toolbox's click simulation, we modified the pipeline to use the ULTRE task Online Service [18] that simulates user clicks based on the rank lists produced by our models during the training process.

4 RESULTS

Table 1 shows the performance of our runs in the ULTRE tasks, including the evaluation metric nDCG@5 and the overall ranking in the the two subtasks for different types of click simulation models.

4.1 Offline ULTR task

In the Offline subtask, we can see that the PRS model has the overall best performance among all the submitted models. It also achieves the best performance for models trained on UCM and MCM click data. While this does not prove that PRS is the best algorithm, it does confirm the algorithm's efficiency for LTR task.

While the ensemble model only ranks third for overall performance, it still achieves the best ranking for models trained on DCM and second-best for UCM. It also outperformed PRS when trained on PBM click data. This is not surprising since the algorithms used for the ensemble model are all state-of-the-art algorithms.

For each type of user click, it seem like ranking models trained on UBM and DCM have the highest performance.

4.2 Online ULTR task

The result shows that DBGD achieved the overall best performance among the three runs we submitted and achieved second highest performance among all the runs when trained on MCM click data. This is unexpected, considering MGD and NSGD usually have a better performance than DBGD. We hypothesize that for the ULTRE task Online API service, each team only has a certain amount of times to call the API; we split the number of calls evenly among the three algorithms. Since DBGD only have two rank list to compare for every training step, it only needs to call the API twice. On the other hand, NSGD and MGD required multiple calls to the API for every training step. Thus, DBGD effectively has more training data than the other two algorithms, and hence, the better performance.

Between Online and Offline ULTR algorithms, to our expectation, Offline algorithms have overall better performance than Online algorithms.

5 CONCLUSIONS

In the NTCIR-16 ULTRE task, we participated in both Online and Offline ULTR subtasks. We implemented an ensemble model for the Offline ULTR task, consisting of five algorithms, IPW, DLA, REM, PairD, PRS trained on an MLP and SetRank, and a separate PRS model trained on SetRank. We used three algorithms for the Online ULTR subtasks, DBGD, NSGD and MGD trained on SetRank. In the future, we would like to utilize a different approach and use Reinforcement Learning to Rank.

REFERENCES

[1] Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and W Bruce Croft. 2018. Unbiased learning to rank with unbiased propensity estimation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 385–394.

[2] Qingyao Ai, Tao Yang, Huazheng Wang, and Jiaxin Mao. 2021. Unbiased learning to rank: online or offline? *ACM Transactions on Information Systems (TOIS)* 39, 2 (2021), 1–29.

[3] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An Experimental Comparison of Click Position-Bias Models. In *Proceedings of the 2008 International Conference on Web Search and Data Mining* (Palo Alto, California, USA) *(WSDM '08)*. Association for Computing Machinery, New York, NY, USA, 87–94. https://doi.org/10.1145/1341531.1341545

[4] Ziniu Hu, Yang Wang, Qu Peng, and Hang Li. 2019. Unbiased lambdamart: an unbiased pairwise learning-to-rank algorithm. In *The World Wide Web Conference*. 2830–2836.

[5] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2017. Accurately Interpreting Clickthrough Data as Implicit Feedback. *SIGIR Forum* 51, 1 (aug 2017), 4–11. https://doi.org/10.1145/3130332.3130334

[6] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. 2007. Evaluating the Accuracy of Implicit Feedback from Clicks and Query Reformulations in Web Search. *ACM Trans. Inf. Syst.* 25, 2 (apr 2007), 7–es. https://doi.org/10.1145/1229179.1229181

[7] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. 781–789.

[8] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*. PMLR, 3744–3753.

[9] Liang Pang, Jun Xu, Qingyao Ai, Yanyan Lan, Xueqi Cheng, and Jirong Wen. 2020. Setrank: Learning a permutation-invariant ranking model for information retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 499–508.

[10] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks: estimating the click-through rate for new ads. In *Proceedings of the 16th international conference on World Wide Web*. 521–530.

[11] Anne Schuth, Harrie Oosterhuis, Shimon Whiteson, and Maarten de Rijke. 2016. Multileave gradient descent for fast online learning to rank. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. 457–466.

[12] Anh Tran, Tao Yang, and Qingyao Ai. 2021. ULTRA: An Unbiased Learning To Rank Algorithm Toolbox. *CoRR* abs/2108.05073 (2021). arXiv:2108.05073 https://arxiv.org/abs/2108.05073

[13] Huazheng Wang, Ramsey Langley, Sonwoo Kim, Eric McCord-Snook, and Hongning Wang. 2018. Efficient exploration of gradient space for online learning to rank. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 145–154.

[14] Nan Wang, Xuanhui Wang, and Hongning Wang. 2020. Unbiased Learning to Rank via Propensity Ratio Scoring. *CoRR* abs/2005.08480 (2020). arXiv:2005.08480 https://arxiv.org/abs/2005.08480

[15] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to Rank with Selection Bias in Personal Search. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Pisa, Italy) *(SIGIR '16)*. Association for Computing Machinery, New York, NY, USA, 115–124. https://doi.org/10.1145/2911451.2911537

[16] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position bias estimation for unbiased learning to rank in personal search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 610–618.

[17] Yisong Yue and Thorsten Joachims. 2009. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*. 1201–1208.

[18] Yurou Zhao, Jiaxin Mao, and Qingyao Ai. 2021. ULTRE framework: a framework for Unbiased Learning to Rank Evaluation based on simulation of user behavior. (2021).