A Rich Ranking Model Based on the Matthew Effect Optimization

Jinzhong Li[1,2](✉) and Guanjun Liu[3,4]

[1] Department of Computer Science and Technology, College of Electronic and Information Engineering, Jinggangshan University, Ji'an 343009, China
[2] Network and Data Security Key Laboratory of Sichuan Province, University of Electronic Science and Technology of China, Chengdu 610054, China
[3] Department of Computer Science and Technology, College of Electronic and Information Engineering, Tongji University, Shanghai 201804, China
[4] Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai 201804, China
1210510@tongji.edu.cn

Abstract. Most existing approaches of learning to rank treat the effectiveness of each query equally which results in a relatively lower ratio of queries with high effectiveness (i.e. rich queries) in the produced ranking model. Such ranking models need to be further optimized to increase the number of rich queries. In this paper, queries with different effectiveness are distinguished, and the queries with higher effectiveness are given higher weights. We modify the gradient in the LambdaMART algorithm based on a new perspective of Matthew effect to highlight the optimization of the rich queries and to produce the rich ranking model, and we present a consistency theorem for the modified optimization objective. Based on the effectiveness evaluation criteria for information retrieval, we introduce the Gini coefficient, mean-variance and quantity statistics to measure the performances of the ranking models. Experimental results show that the ranking models produced by the gradient-modified LambdaMART algorithm based on Matthew effect exhibit a stronger Matthew effect compared to the original LambdaMART algorithm.

Keywords: Learning to rank · Ranking model · Matthew effect LambdaMART algorithm · Gradient

Supported by the National Natural Science Foundation of China under Grant Nos. 61762052 and 61572360, the Natural Science Foundation of JiangXi Province of China under Grant No. 20171BAB202010, the Opening Foundation of Network and Data Security Key Laboratory of Sichuan Province under Grant No. NDSMS201602, the Science and Technology Project of the Education Department of Jiangxi Province of China under Grant No. GJJ160746, and the Doctoral Scientific Research Startup Foundation of Jinggangshan University under Grant No. JZB1804.

1 Introduction

Ranking is an important component that directly affects the performances of information retrieval systems such as search engines and recommendation systems. For instance, the underlying assumption of the PageRank algorithm [1] is that more important websites are likely to receive more links from other websites, it assigns a Web page with higher score if the sum of its corresponding backlinks is high. The PageRank algorithm exhibits the Matthew effect [2] to some extent, which refers to the phenomenon that the rich get richer and the poor get poorer. It is valuable since the PageRank algorithm evaluates the importance of web pages by the link analysis, and ranks web pages by the scores of the importance of web pages. Therefore, the Matthew effect is regarded as a desirable behavior of the ranking model.

Moreover, keywords ranking algorithm of Baidu, goods ranking rules of Taobao, and collaborative filtering algorithms for recommender systems are all showed the Matthew Effect to their respective degrees. Those ranked results of queries or recommendations at higher positions are likely to be the desired target pages of more number of people than those at lower positions, and the ranking of results exhibits the Matthew effect. Furthermore, it is necessary to consider the differences of different queries and to treat those queries distinctly when solving the ranking problem for information retrieval, due to the Matthew effects of the ranking. Therefore, it is a natural idea to distinguish the effectiveness scores of different queries in the training process of the ranking models.

Learning to rank for information retrieval refers to the machine learning techniques in order to train the ranking models in the ranking task. The existing approaches of learning to rank, such as LambdaMART [3,4], CCRank [5], ES-Rank, IESR-Rank and IESVM-Rank [6], Factorized Ranking SVM and Regularized Ranking SVM [7], all equally treat the effectiveness of each query in the optimization process of the ranking models, and these approaches do not distinguish the differences among the effectiveness of different queries. Here, we note that the effectiveness can be measured by any commonly-used information retrieval metrics (e.g. $NDCG$ [8] and ERR [9]). Due to the fact that the effectiveness of different queries may be different for the same ranking model, treating each query equally in the optimization process of the ranking models results in a relatively fewer number of rich queries. To this end, the ranking model needs to be further optimized to further increase the number of rich queries.

In this paper, we modify the gradient in LambdaMART algorithm based on Matthew effect from a new perspective. We describe how the gradient is modified, and present a consistency theorem. Based on the effectiveness evaluation criteria for information retrieval, we introduce the Gini coefficient, mean-variance and quantity statistics to measure the performances of the ranking models. Moreover, we conduct experiments to compare the performances of the ranking models between these models trained by the gradient-modified LambdaMART algorithm based on the Matthew effect (named as Matthew-λ-MART) and those models trained by the original one. The experimental results indicate that the Matthew-λ-MART exhibits a stronger Matthew effect.

2 Construction of the Rich Ranking Models via Matthew Effect

2.1 The Gradient of LambdaMART Algorithm

LambdaMART [3,4] is a state-of-the-art learning to rank algorithm, which has been proven to be very successful in solving real world ranking problems. An ensemble of LambdaMART rankers won the “2010 Yahoo! Learning to Rank Challenge”.

The main feature of the LambdaMART [3,4] algorithm is the definition of the gradient function λ of the loss function without directly defining the loss function. λ quantifies the force of a ‘to-be-sorted’ document and points out the upward or downward adjustment direction in the next iteration. The two documents in each document pair are associated with a query have different relevance, and the gradients of the two documents are equivalent but their moving directions are opposite to each other. The gradient of the positive direction pushes the document toward the top of the ranked list, while the gradient of the negative direction pushes the document toward the bottom of the ranked list.

The LambdaMART [3,4] algorithm optimizes the gradient λ_i of each document d_i for each query q to train the ranking models. If the relevance judgement r_i between d_i and q is higher, and the ranked position of d_i is closer to the bottom of the ranked list for a given query q, then the positive value of λ_i indicates a push toward the top of the ranked list. Meanwhile, if the value of λ_i is bigger, then it shows that the force is stronger. If the relevance r_i between d_i and q is smaller, and the ranked position of d_i is closer to the top of the ranked list, then the negative value of λ_i indicates a push toward the bottom of the ranked list. Meanwhile, if the value of λ_i is smaller, then it shows that the force is stronger.

The LambdaMART [3,4] algorithm integrates the evaluation criteria ($NDCG$) of the information retrieval into the computation of the gradient. The gradient λ_i for each document d_i is obtained by summation of all λ_{ij} over all pairs of $<d_i, d_j>$ that d_i participates in for query q. Therefore, λ_i can be written as $\lambda_i = \sum_{j:\{i,j\}\in I} \lambda_{ij} - \sum_{j:\{j,i\}\in I} \lambda_{ij}$, where $\lambda_{ij} = -\frac{\beta}{1+e^{\beta\times(s_i-s_j)}} \times |\Delta M_{ij}|$, which denotes the gradient of the document pair $<d_i, d_j>$. In the formula $\lambda_{ij} = -\frac{\beta}{1+e^{\beta\times(s_i-s_j)}} \times |\Delta M_{ij}|$, β is a shape parameter for the sigmoid function, s_i and s_j represent the score assigned to d_i and d_j by the ranking model respectively, and ΔM_{ij} represents the change on effectiveness measure M by swapping the two documents d_i and d_j at rank positions i and j accordingly (while keeping the rank positions of all other documents unchanged). Therefore, ΔM_{ij} can be calculated by $\Delta M_{ij} = M_q - M_q^*$, where M_q denotes the effectiveness of query q for a ranked list of all documents w.r.t. q, M_q^* denotes the effectiveness of query q after swapping the documents d_i and d_j at the rank positions i and j for the ranked list, and M denotes the effectiveness evaluation criterion.

2.2 Modification of the Gradient of the LambdaMART Algorithm

The LambdaMART algorithm equally treats the effectiveness of each query in the training process of the ranking models. In order to discriminate the effectiveness score of each query, we assign different weights to different queries with unequal effectiveness scores for optimizing the gradients in the training process of the ranking models. In order to highlight the gradients of rich queries and enhance the effectiveness of rich queries, the weights of rich queries should be given the higher values based on the idea of the Matthew effect. We assign the effectiveness score of each query as the weight of the corresponding query. Therefore, the original effectiveness M is replaced by the new objective M^2 to modify the gradient of the original LambdaMART algorithm, thereby expanding the LambdaMART algorithm to optimize the rich ranking model. Therefore, ΔM_{ij} is replaced by ΔM_{ij}^* when λ_{ij} is computed, where $\Delta M_{ij} = M_q - M_q^*$ and $\Delta M_{ij}^* = (M_q)^2 - (M_q^*)^2$, which denotes the difference of the squared effectiveness scores of query q after swapping the two documents d_i and d_j at rank positions i and j (while keeping the rank positions of all other documents unchanged). In other words, $M_q - M_q^*$ is replaced by $(M_q)^2 - (M_q^*)^2$. The document pair $<d_i, d_j>$ for the same query is optimized according to the new gradient $\lambda_{ij}^{new} = -\frac{\beta}{1+e^{\beta\times(s_i-s_j)}} \times |\Delta M_{ij}^*|$, which strengthens the differences of upward or downward ranking force among the document pairs for different queries in the next iteration, and thus enhances the optimization of rich queries.

The gradient function of the LambdaMART algorithm is modified, and therefore, it is necessary to demonstrate that the Matthew-λ-MART algorithm can be used to train the ranking model for the learning to rank task. Now, we present that the new optimization objective M^2 satisfies the consistency property proposed in [4]: when swapping the ranked positions of two documents, d_i and d_j, in a ranked list of documents where d_i is more relevant than d_j but d_i is ranked after d_j, the optimization objective should be increased. In other words, for any document-pairs, the pairwise swap between correctly ranked documents d_i and d_j for the same query q must lead to a decrease of M^2, and the pairwise swap between the incorrectly ranked documents d_i and d_j for the same query q must lead to an increase of M^2.

Theorem 1. The new optimization objective M^2 satisfies the consistency property.

3 Evaluation Measures of Matthew Effect

A ranking model is richer than another model if (1) the former both has more rich queries and has more poor queries than the latter, or (2) the distribution of the effectiveness of queries of the former is more discrete than the latter. A richer ranking model is of a stronger Matthew effect. In order to measure the performances of the ranking model yielded by our modified approach, we introduce the following utility metrics to characterize the Matthew effect of ranking models from different perspectives. Based on the effectiveness evaluation criteria for information retrieval, we introduce Gini coefficient, mean-variance and quantity statistics to measure the performances.

3.1 Gini Coefficient

The Gini coefficient [10] is a measure of the statistical dispersion, which is intended to represent the income distribution of the residents in a nation, and is the commonly used as a measure of inequality of income or wealth. The Gini coefficient is calculated using the formula $G = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}|x_i - x_j|}{2n^2\mu}$, where x_i denotes the income of individual i, $|x_i - x_j|$ denotes the absolute value of the difference between x_i and x_j, μ denotes the mean value of all individuals' incomes, and n denotes the total number of individuals. The smaller the inequality of income, the smaller the value of the Gini coefficient; and vice versa.

The Matthew effect is reflected by using Gini coefficients for the measurement in many economic areas, so Gini Coefficient can capture the Matthew Effect, which can be used to measure the performance of the ranking model. If we make an analogy with the distribution of the national income in the field of finance, the query in the learning to rank task resembles the individual in the distribution of the national income, and the effectiveness of the query resembles the income of the individual. Therefore, the Gini coefficient of learning to rank can be defined as follows:

$$Gini = \frac{\sum_{i=1}^{|Q|}\sum_{j=1}^{|Q|}|M_{q_i} - M_{q_j}|}{2|Q|\sum_{k=1}^{|Q|}M_{q_k}} \tag{1}$$

where, $q_i \in Q$ denotes the i-th query and $|Q|$ represents the total number of queries in query set Q.

Gini is used to measure the degree of difference in effectiveness among all the queries in a ranking model, and to reflect the Matthew effect by comparing the *Gini* of one ranking model with another. A higher Gini value represents a greater difference in effectiveness (i.e. effectiveness inequality) among all the queries in a ranking model. If the value of *Gini* obtained by a ranking model is larger, then it indicates that the corresponding ranking model has a stronger Matthew effect.

3.2 Mean-Variance

In probability theory and mathematical statistics, mean is used to measure the average value of all random variables; Variance is used to measure the degree of deviation between a set of random variables and their mean, and it is an important and commonly used metric for calculating the discrete trend.

In order to observe the effectiveness of a ranking model, the mean μ of a ranking model is defined as follows:

$$\mu = \frac{1}{|Q|}\sum_{q \in Q} M_q \tag{2}$$

For a ranking model, the mean μ measures the average effectiveness (such as *NDCG* and *ERR*) of all queries in a set of queries, i.e., it refers to the average effectiveness of the ranking model. The greater the mean μ, the better the average effectiveness of the ranking model; and vice versa.

In a ranking model, some queries are of high effectiveness but some are of low effectiveness. Therefore, in order to observe the degrees of their deviation, we divide the variance of a ranking model into the upside semi-variance V_{up} and the downside semi-variance V_{down}, which are defined as follows:

$$V_{up} = \frac{1}{|Q^+|} \sum_{q \in Q^+} (M_q - \mu)^2 \tag{3}$$

$$V_{down} = \frac{1}{|Q^-|} \sum_{q \in Q^-} (M_q - \mu)^2 \tag{4}$$

where, Q^+ and Q^- denote the set of queries with above-mean effectiveness and below-mean effectiveness in the query set Q respectively, and $|Q^+|$ and $|Q^-|$ denote the number of the set of queries Q^+ and Q^- respectively. The greater the variance, the greater the degree of deviation; and vice versa.

For a ranking model, the V_{up} and the V_{down} measures the discrete degree of effectiveness of the queries that are over and under the μ in the query set respectively. The Matthew effect of a ranking model is exhibited by comparing the V_{up} and the V_{down} of the ranking model to those of other ranking models respectively. If the values of both V_{up} and V_{down} of a ranking model are higher, then it indicates that the ranking model has a stronger Matthew effect; and vice versa.

3.3 Quantity Statistics

The range of values of the most commonly used effectiveness measures is between 0 and 1 in information retrieval, such as *NDCG* and *ERR*. To compute the effectiveness distribution of all queries in a query set, the range of values of the effectiveness is divided into 5 intervals as $[0.0, 0.2]$, $(0.2, 0.4]$, $(0.4, 0.6]$, $(0.6, 0.8]$ and $(0.8, 1.0]$, respectively. We compute the number of queries distributed in these different intervals according to the effectiveness values of the queries in a given query set for different ranking models, and the purpose is to evaluate the strengths of their exhibited Matthew effect. We use an array *count* to express the quantity statistics of different intervals for the effectiveness of queries, and the *count* is defined as follows:

$$count[i] = \begin{cases} \sum_{M_q \in [0.0, 0.2], q \in Q} 1 & i = 0 \\ \sum_{M_q \in (0.2, 0.4], q \in Q} 1 & i = 1 \\ \sum_{M_q \in (0.4, 0.6], q \in Q} 1 & i = 2 \\ \sum_{M_q \in (0.6, 0.8], q \in Q} 1 & i = 3 \\ \sum_{M_q \in (0.8, 1.0], q \in Q} 1 & i = 4 \end{cases} \tag{5}$$

If the values of $count[0]$ and $count[4]$ obtained by a ranking model are larger, then the ranking model has a stronger Matthew effect.

4 Experiments

In order to verify the performances of the Matthew-λ-MART, we implement the algorithm based on the open-source RankLib library of learning to rank algorithms developed by Van Dang et al.[1] Based on the effectiveness measures $NDCG$ and ERR, we conduct experiments on Microsoft Learning to Rank dataset MSLR-WEB30K[2], which is the larger scale dataset of learning to rank and makes it possible to derive reliable conclusions. We report the total results of all five folds for the test dataset. The utility metrics used in our experiments are μ, V_{up}, V_{down}, $Gini$ and $count$ respectively, and their results are shown in Figs. 1, 2, 3, 4, 5 and 6.

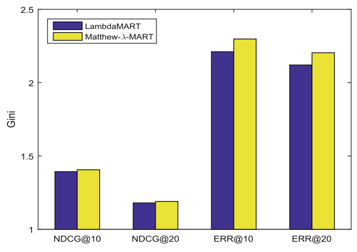

Fig. 1. *Gini* of each algorithm on MSLR-WEB30K dataset

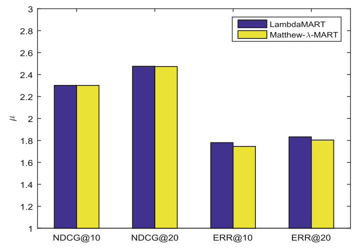

Fig. 2. μ of each algorithm on MSLR-WEB30K dataset

[1] http://sourceforge.net/p/lemur/code/HEAD/tree/RankLib/trunk/.
[2] http://research.microsoft.com/en-us/projects/mslr/download.aspx.

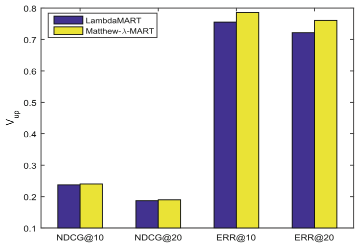

Fig. 3. V_{up} of each algorithm on MSLR-WEB30K dataset

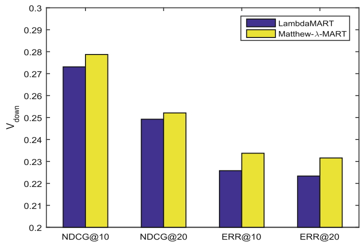

Fig. 4. V_{down} of each algorithm on MSLR-WEB30K dataset

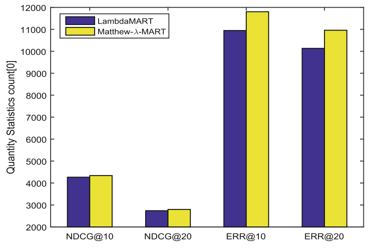

Fig. 5. $count[0]$ of each algorithm on MSLR-WEB30K dataset

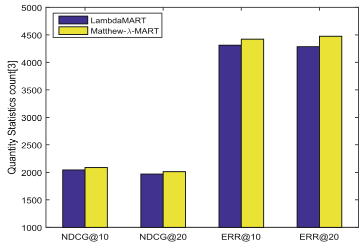

Fig. 6. $count[4]$ of each algorithm on MSLR-WEB30K dataset

From the perspectives of the Gini coefficient in Fig. 1, Matthew-λ-MART obtains the bigger *Gini* than LambdaMART on all effectiveness measures (including $NDCG$@10, $NDCG$@20, ERR@10 and ERR@20). These results show that the effectiveness across different individual queries in Matthew-λ-MART has a greater difference. Therefore, the ranking models trained by Matthew-λ-MART exhibit a stronger Matthew effect about *Gini*.

From the perspectives of mean-variance in Figs. 2, 3 and 4, although the μ obtained by the Matthew-λ-MART are smaller, the corresponding V_{up} and V_{down} are both bigger than the LambdaMART on all the above effectiveness measures. These results show that the effectiveness across different individual queries in Matthew-λ-MART has also a greater difference. Therefore, the ranking models trained by Matthew-λ-MART also exhibit a stronger Matthew effect about V_{up} and V_{down}.

From the perspectives of quantity statistics in Figs. 5 and 6, the $count[4]$ obtained by the Matthew-λ-MART w.r.t. rich queries and the corresponding $count[0]$ w.r.t. poor queries are both bigger than LambdaMART on all above effectiveness measures. These results of Matthew-λ-MART produce a relative polarization. Therefore, the ranking models trained by Matthew-λ-MART further exhibit a stronger Matthew effect about $count[0]$ and $count[4]$.

The primary reason for the above observations is that the gradient is modified by the Matthew effect in the original LambdaMART algorithm. Matthew-λ-MART highlights the corresponding differences of upward or downward ranking force between documents w.r.t. rich queries with high effectiveness and documents w.r.t. poor queries with low effectiveness in the next iteration. Therefore, the optimization of rich queries is strengthened and the optimization of poor queries is weakened accordingly. Therefore, more attentions are paid to optimize the ranked positions of the documents in the rich queries while less attentions for the poor queries. It leads to an increase in the corresponding numbers of both the rich queries and the poor queries respectively. Finally it increases the diversion or degree of difference in the effectiveness across all the queries in the ranking models.

5 Conclusion

To highlight the high effectiveness of the important queries and to abandon the average effectiveness across all the queries, the queries with different effectiveness are treated distinctly in our proposed approach, and they are assigned with different weights. Based on the new perspectives of Matthew effect, we modify the gradient in the LambdaMART algorithm by assigning higher weights for the gradients of the queries with higher effectiveness so as to highlight the optimization of these rich queries, and thereby produce the rich ranking model. We introduce the Gini coefficient, mean-variance, and quantity statistics to quantize the Matthew effect of the ranking models. In comparison with the original LambdaMART algorithm, the ranking models trained by the gradient-modified LambdaMART algorithm based on the Matthew effect exhibits a stronger Matthew effect.

It is obvious that different information has different popularity in different time periods, which causes the popularity of queries to change over time. Some of the queries (hot queries) gain a huge popularity with numerous searchers, while some of the queries (cold queries) are just opposite. Most existing approaches of learning to rank treat all the queries with equal weights and the popularity factor of the queries is neglected. Therefore, the hot queries and the cold queries are not treated differently. If the hot queries are not treated with higher priorities, then the huge number of users searching such hot queries may not be satisfied, which will degrade the overall user experiences. In order to increase the quality of user experiences, more weights should be assigned to the hot queries during the training process of the ranking models. As a future work, we plan to integrate the hot queries and the cold queries into the Matthew-effect-based gradient-modified LambdaMART algorithm to construct the ranking models. To make the rank of search results of hot query more effective, we will give more weights to the hot queries and less weights to the cold queries in the training process of the ranking models, so as to enhance the overall user experiences.

References

1. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: bringing order to the web. Technical report, Stanford Digital Library Technologies Project (1999)
2. Merton, R.K., et al.: The matthew effect in science. Science **159**(3810), 56–63 (1968)
3. Wu, Q., Burges, C.J., Svore, K.M., Gao, J.: Adapting boosting for information retrieval measures. Inf. Retr. **13**(3), 254–270 (2010)
4. Burges, C.J.: From ranknet to lambdarank to lambdaMART: an overview. Microsoft Research Technical report MSR-TR-2010-82 (2010)
5. Wang, S., Wu, Y., Gao, B.J., Wang, K., Lauw, H.W., Ma, J.: A cooperative coevolution framework for parallel learning to rank. IEEE Trans. Knowl. Data Eng. **27**(12), 3152–3165 (2015)
6. Ibrahim, O.A.S., Landasilva, D.: An evolutionary strategy with machine learning for learning to rank in information retrieval. Soft Comput. **22**(10), 3171–3185 (2018)
7. Xu, J., Zeng, W., Lan, Y., Guo, J., Cheng, X.: Modeling the parameter interactions in ranking SVM with low-rank approximation. IEEE Trans. Knowl. Data Eng. (2018, in Press)
8. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. ACM Trans. Inf. Syst. (TOIS) **20**(4), 422–446 (2002)
9. Chapelle, O., Metlzer, D., Zhang, Y., Grinspan, P.: Expected reciprocal rank for graded relevance. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, pp. 621–630. ACM (2009)
10. Dixon, P.M., Weiner, J., Mitchell-Olds, T., Woodley, R.: Bootstrapping the gini coefficient of inequality. Ecology **68**, 1548–1551 (1987)