Journal of Engineering and Technology for Industrial Applications

ISSN ONLINE: 2447-0228

ITEGAM-JETIA

Manaus, v.12 n.58, p 829-834. March/April, 2026.
DOI: https://doi.org/10.5935/jetia.v12i58.3090

RESEARCH ARTICLE OPEN ACCESS

PERSONALIZED INFORMATION RETRIEVAL USING LETOR MACHINE LEARNING RE-RANKING ALGORITHMS IN MSLRWEB10K DATASET: A COMPREHENSIVE STUDY

Mrs. M. Poomani*[1] and Dr. J. Jebamalar Tamilselvi[2]

[1]Research Scholar, Department of Computer Science, FSH, SRM Institute of Science and Technology, Ramapuram, Chennai, Tamilnadu, India
[2]Associate Professor, Department of Cyber Security , SRM Institute of Science and Technology, Ramapuram, Chennai, Tamilnadu, India

[1]http://orcid.org/0009-0006-8918-6877, [2]http://orcid.org/0009-0004-1292-1293

Email: *poomanim@srmist.edu.in, jebamalj@srmist.edu.in

ARTICLE INFO

Article History
Received: December 9, 2025
Reviewed: January 10, 2026
Accepted: March 10, 2026
Published: April 30, 2026

Keywords:
Personalized Information Retrieval
Machine Learning
Re-Ranking Algorithms
User Modeling
Search Personalization

ABSTRACT

Personalized Information Retrieval (PIR) aims to tailor search results to individual user preferences and contexts. With the exponential growth of digital information, traditional retrieval systems often fall short in delivering relevant results to diverse users. This study explores the integration of machine learning re-ranking algorithms into personalized information retrieval systems to enhance search relevance and user satisfaction. The LETOR based model is experimented for relevance re-ranking in personalized retrieval. A comprehensive analysis of LETOR based models are analyzed to find a best hybrid re-ranking framework based on the performance of each model. The findings demonstrate that the LGBMRegressor model demonstrated the most consistent and best performance across the majority of metrics.

I. INTRODUCTION

The digital age has ushered in an era of information overload, making efficient and relevant information retrieval a critical challenge. Personalized Information Retrieval (PIR) addresses this by customizing search results based on individual user preferences, behaviors, and contexts. Traditional PIR approaches often rely on static user profiles or heuristic-based methods, which may not adapt well to dynamic user needs. Recent advancements in machine learning have introduced re-ranking algorithms that can dynamically adjust search results based on learned user preferences. These algorithms offer the potential to significantly enhance the relevance and personalization of search outcomes. Machine learning personalizes search results for individual users by continuously analyzing and learning from a wide range of user data and behavioral patterns.

II. RELATED WORK

The methodology in the paper "Incorporating Explicit Subtopics in Personalized Search" (ExpliPS model) introduces a three-stage architecture to enhance personalized search by explicitly modeling user preferences for subtopics of a query. It uses BERT to generate dense vector representations for these documents as well the queries. The Final score is calculated by the weighted combination of document-query relevance and user-subtopic preference. The model is trained using a pairwise ranking loss, such as Bayesian Personalized Ranking (BPR), to ensure preferred documents are ranked higher than non-preferred ones [1].

Traditional personalized search models often rely on static representations of user interests derived from historical behaviors, such as past queries and clicked documents. These static profiles may not effectively capture the dynamic nature of user intent, leading to less accurate personalization. Moreover, documents are typically lengthy and encompass multiple topics, making fixed-length representations insufficient and potentially introducing noise into the user profile. To address these challenges, the author propose a method that constructs dynamic, intent-oriented representations of documents. The author suggest to use cross-attention or similarity-based matching to measure the alignment between current candidate documents and the user's evolving interests to enhance relevance estimation by capturing subtle shifts in intent.

Training uses a pairwise learning-to-rank loss, such as BPR or hinge loss, encouraging more relevant documents to rank higher than less relevant ones [2]. Traditional neural relevance models often overlook the nuanced context of user queries, such as geographical location or professional role. This omission can lead to less personalized and relevant search results. Kong, D., Zhou, D., Huang, Z., & Sigalas, S. aims to address this gap by developing a model that incorporates both the content of documents and the contextual information associated with user queries.

The model uses a hybrid approach that fuses: Lexical Matching Features with Neural Semantic Relevance Features and Learning-to-Rank Layer. These features aim to capture deeper semantic compatibility between user intent and document meaning. Contextual Feature Representation model builds a composite user profile vector from various contextual signals. Each context field is embedded separately and then aggregated (e.g., via average or attention). The Learning-to-Rank (LTR) Layer is typically a feed forward neural network or gradient-boosted tree (GBT) model. Models incorporating context (e.g., title-location-query-document matching) consistently outperformed baseline models [3].

III. METHODOLOGY

III.1 PERSONALIZED SEARCH TECHNIQUES

Initial personalized information systems mainly relied on direct user feedback and fixed profiles to customize search outcomes. Nevertheless, these approaches frequently demonstrated a deficiency in their ability to adjust to evolving user preferences. Recent studies have investigated dynamic user modeling, integrating real-time behaviors and context-related data to enhance personalization strategies. Machine learning has transformed information retrieval by allowing systems to learn from data and enhance their performance over time. Re-ranking algorithms, in particular, have showed potential in changing the first search results to better fit what the user wants.

Methods like Learning to Rank (LTR) have gained significant traction, utilizing features derived from user interactions and the relevance of content [4].Combining user modeling with machine learning re-ranking has become a powerful way to get personalized information. Hybrid models effectively integrate long-term user preferences with immediate contextual cues, resulting in search results that are both accurate and personalized. Here is an overview of several commonly employed techniques:

III.2 COLLABORATIVE FILTERING

Collaborative filtering suggests content by analyzing the similarities among users. When individuals with comparable profiles express a preference for specific outcomes, it increases the likelihood that those outcomes will be presented to others within that same demographic. These systems offer the benefit of addressing the cold-start problem to a certain degree. Handling extensive datasets and determining user similarities can be resource-intensive and may struggle to adjust to evolving user preferences. [5].

III.3 CONTENT-BASED FILTERING

Content-based filtering emphasizes the attributes of items that a user has interacted with in the past, suggesting analogous content for future engagement. It constructs comprehensive user profiles by monitoring search queries, clicks, time spent on pages, and engagement patterns over time. This historical data enables the system to comprehend individual user interests and preferences, thereby customizing future outcomes accordingly. This approach effectively addresses the issue of initial data scarcity by providing recommendations despite minimal user interaction information. The system is capable of managing substantial volumes of data with high efficiency, rendering it appropriate for extensive online platforms. [6-8].

III.4 HYBRID METHODS

Hybrid models present a compelling strategy for enhancing personalization in machine learning, facilitating the development of systems that are more precise, resilient, and flexible. Integrating the advantages of various models allows for the tackling of personalization challenges and enhances user experiences [9-12].

III.5 CONTEXT-AWARE PERSONALIZATION

When offering recommendations or changing services, context-aware personalization models take into account the user's present condition and environment. These models utilize real-time data, including location, device type, time of day, and the user’s current session behavior, to deliver more relevant and customized experiences. A search for “leagues” might produce sports-related results if the user frequently accesses sports websites, or it could vary depending on geographic location or the time of the search [13].

III.6 REINFORCEMENT LEARNING

Personalization through reinforcement learning (RL) serves as an effective mechanism, enabling systems to evolve and understand user preferences progressively. By considering each user as an agent engaging with their surroundings, RL has the potential to tailor user experiences across multiple domains. Personalization driven by reinforcement learning presents a compelling method for customizing user experiences according to specific preferences and requirements. Through ongoing analysis of user interactions, RL systems are able to deliver increasingly relevant, engaging, and tailored services. Nonetheless, it is crucial to tackle the obstacles related to data needs, computational capabilities, and ethical implications [14].

III.7 NEURAL AND EMBEDDING-BASED PERSONALIZATION

It employs deep learning to learn user and item representations (for example, through embedding's). Embedding models serve as a category of machine learning models that aim to represent various forms of data, including text and images, within a continuous, low-dimensional vector space.

These embedding's capture semantic or contextual similarities between pieces of data, allowing machines to execute tasks such as comparison, clustering, or classification with greater efficiency. These models transform words into numerical vectors that encapsulate semantic meanings and the relationships that exist between them. Models such as Word2vec employ straightforward neural networks to acquire embedding's grounded in co-occurrence patterns. Transformers, such as BERT and GPT, along with CNNs, are adept at extracting intricate patterns and relationships through layered data processing. [15].

III.8 LEARNING-TO-RANK FOR PERSONALIZATION

Learning to Rank (LTR) is a type of machine learning that changes the order of items in a list based on how relevant or important they are. This is very helpful for making recommendations and search engines more personalized. The system analyzes user interactions and various data sources to forecast the optimal ranking order, thereby enhancing the personalization and relevance of the results. LTR models can be developed through a range of techniques, such as point wise, pairwise, and list wise approaches. Learning to Rank serves as an effective mechanism for personalization, allowing systems to provide users with more pertinent and engaging outcomes by analyzing their interactions and behaviors [16-20].

IV. PROPOSED METHODOLOGY

For any user query, a relevance function can take into account any one or multiple factors such as:

1. Text similarity
2. Query properties
3. Document properties
4. User and context properties

When considering a system based on the user and context properties we can derive any one of the systems like: General Traditional Search, Personalized Search and Recommendation systems. When considering personalized search both user input (query) and user preferences are part of the relevance parameter. This implies that search personalization should be implemented with utmost care. The search may end up annoying or be biased to users with their favorite documents while they were particularly looking for something else if the emphasis was on historical user behavior and too little on the current search intent. This illustrates the importance of providing users with enough historical data to confidently tilt search results in a particular way.

Additionally, remember that personalization will mostly impact exploratory inquiries and confusing user input. The search algorithms should already cover clear navigational inquiries. Learning-to-rank (LTR) is the process of creating statistical models for relevance ranking. The model is trained to identify the best options from certain data, rather than manually creating a structured query and weights for all text similarity, query attributes, and document properties. The information is presented as a decision list. The behavior-based personalization is examined with LTR, which means that the user attributes is extracted and will be utilized in our LTR training process by leveraging historical user activity.

An initial set of search results is generated using a standard retrieval model, such as BM25, based on the user's query. The Microsoft Learning to Rank (MSLR) dataset MSLR-WEB10K is used to conduct the experiment. These datasets are used in learning to Rank (LTR) problems, where each instance (document) is associated with a query, a set of features, and a relevance label. The baseline ranking score is based on the original document order, without using any model and the predicted ranking scores is calculated.Then the following hybrid re-ranking framework is trained and tested that integrates user profiling with machine learning algorithms to enhance Personalized Information Retrieval.

- LightGBM (using LGBMRanker).
- LGBMRegressor (using LambdaRank)
- XGBoost-based Learning-to-Rank model (using XGBRanker)

For Learning to Rank models like LightGBM's ranker, the model needs no of documents for each query as input. This is done using group info (an array) where each value is the number of documents for one query. Then the unique query IDs and the number of times each appears are calculated and found (i.e., how many documents are associated with each query).Then the counts (documents per query) is extracted, which is used as training group or testing group. When training ranking models, rankings per query is evaluated, so Group documents are done correctly by query. The model must know how many documents are associated with each query. So we now load the MSLR training/testing data and separate the features, labels, and query IDs.

Then calculate how many documents belong to each query (group info). This is essential input for Learning-to-Rank models that optimize rankings on a per-query basis. Then the documents are sorted by predicted score (descending), and the same order is applied to simulate the ranked list. ndcg_score needs relevance and ideal relevance values. So the actual relevance list based on predicted ranking is calculated and found. The ideal ranking scores is simulated, assuming perfect prediction order of k=10 to evaluate top-10 results only. Then the true relevance score is converted into a binary vector considering (≥1) for relevant, 0 for irrelevant. The average_precision_score() computes precision across all ranks and it adds score if there's at least one relevant document. Then find the index of the first relevant document (relevance ≥1). Use the formula [1 / (rank of first relevant doc)] for finding the MRR score and then find the average of each metric across all queries. Learning to Rank models on multi-query datasets is evaluated for the dataset.

Then the training and testing is done using LightGBM's LGBMRanker. LGBMRegressor is a regression model, not specifically designed for ranking. It's commonly used as a point wise LTR baseline, where the model predicts relevance scores directly. This ranking approach comes under the point wise approach, where each document is scored independently. This method ignores query groups, meaning it doesn't model interactions between documents within the same query. It's less effective than pairwise or list wise models like LGBMRanker (LambdaRank), but useful as a simple baseline. Then we train for the XGBoost-based Learning-to-Rank model (XGBRanker) using the NDCG objective, and applies it to predict document ranking scores. It's similar in purpose to LGBMRanker, but uses the XGBoost instead of LightGBM. Learning to Rank (LTR) algorithm is employed, such as LambdaMART, to re-rank the initial search results. The model is trained on labeled data to predict the likelihood of a document being relevant to the user.

V. EXPERIMENT

V.1 DATASET

The testing of the framework is done using the MSLRWEB10K dataset, which includes user queries, click-through data, and relevance judgments. It consist of 136 features with 10K Queries in each fold with a total count of 5 folds.

V.2 FEATURES

For each retrieved document, we extract a set of features, including:

- Query-document relevance scores
- User-document interaction history
- Temporal and contextual factors

V.3 BASELINE MODELS

The retrieval models evaluated are:

- Baseline ranking quality based only on the original document order.
- LightGBM (using LGBMRanker).
- LGBMRegressor (using LambdaRank)
- XGBoost-based Learning-to-Rank model (using XGBRanker)

V.4 EVALUATION METRICS

Performance was assessed using standard IR metrics as mentioned below:

- **Mean Reciprocal Rank (MRR) -** measures the average reciprocal rank of the first relevant document for a given query. MRR is calculated as

$$RR = 1/Q * \Sigma (1/rank\_q),$$

Where Q is the number of queries and rank_q is the rank of the first relevant result for query q.

- **Normalized Discounted Cumulative Gain (NDCG) -** Normalized Discounted Cumulative Gain (NDCG) is a more comprehensive metric that considers both the relevance of retrieved documents and their position in the ranked list.

$$NDCG@10 = DCG@10 / IDCG@10$$

DCG@10 is the Discounted Cumulative Gain for the top 10 results. IDCG@10
Is the Ideal Discounted Cumulative Gain for the top 10 results.

- **Mean Average Precision (MAP) -** Mean Average Precision (mAP) is calculated by averaging the Average Precision (AP) across all classes or categories in a dataset.

$$mAP = (1/n) * sum (AP),$$

Where 'n' is the number of classes or categories.

VI. RESULTS

The graph depicts the result for all three re-ranking techniques for MSLRWEB10K dataset by having the relevancy score on the y-axis for each Metrics for all the three re-ranking techniques along with original order without re-ranking

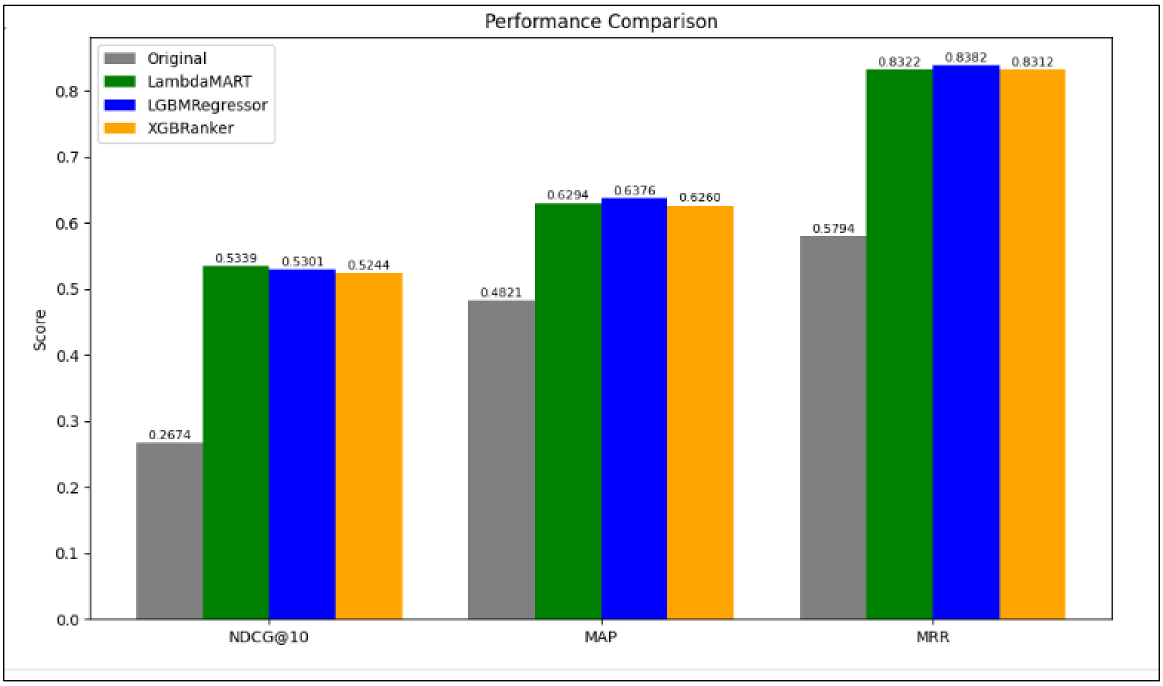

Fig 1. Performance Comparison of Proposed Model based on MSLRWEB10K dataset.
Source: Authors, (2026).

VII. CONCLUSION

This study demonstrates the efficiency of Re-ranking models and its performance. Among the evaluated models, LGBMRegressor demonstrated the most consistent and best performance across the majority of metrics. It achieved the highest scores in both MAP (0.6376) and MRR (0.8382), and also gives good performance in NDCG@10 (0.5301). Although LambdaMART slightly outperformed in NDCG@10, the overall results indicate that LGBMRegressor offers a better balance of ranking quality and retrieval effectiveness, making it the most effective model in this comparison.

VIII. AUTHOR'S CONTRIBUTION

Conceptualization: Mrs. M. Poomani and Dr. J. Jebamalar Tamilselvi.
Methodology: Mrs. M. Poomani and Dr. J. Jebamalar Tamilselvi.
Investigation: Mrs. M. Poomani and Dr. J. Jebamalar Tamilselvi.
Discussion of results: Mrs. M. Poomani and Dr. J. Jebamalar Tamilselvi.
Writing – Original Draft: Mrs. M. Poomani and Dr. J. Jebamalar Tamilselvi.
Writing – Review and Editing: Mrs. M. Poomani and Dr. J. Jebamalar Tamilselvi.
Resources: Mrs. M. Poomani and Dr. J. Jebamalar Tamilselvi.
Supervision: Mrs. M. Poomani and Dr. J. Jebamalar Tamilselvi.
Approval of the final text: Mrs. M. Poomani and Dr. J. Jebamalar Tamilselvi.

IX.REFERENCES

[1] Wang, S., Yao, J., Zhou, Y., Dou, Z., & Wen, J. (2023). Incorporating Explicit Subtopics in Personalized Search. In Proceedings of the ACM Web Conference 2023. https://doi.org/10.1145/3543507.3583488

[2] Bai, Y., Zhou, Y., Dou, Z., & Wen, J. (2024). Intent-Oriented Dynamic Interest Modeling for Personalized Web Search. ACM Transactions on Information Systems, 42(4), 1-30. https://doi.org/10.1145/3639817

[3] Kong, D., Zhou, D., Huang, Z., & Sigalas, S. (2023). Personalized Search Via Neural Contextual Semantic Relevance Ranking. ArXiv preprint arXiv: 2309.05113. https://arxiv.org/abs/2309.05113

[4] Zhou, Y., Dou, Z., Wei, B., Xie, R., & Wen, J.-R. (2021). Group Based Personalized Search by Integrating Search Behaviour and Friend Network. arXiv preprint arXiv:2111.12618. https://arxiv.org/abs/2111.12618

[5] Papadakis H, Papagrigoriou A, Panagiotakis C, Kosmas E, Fragopoulou P. Collaborative filtering recommender systems taxonomy. Knowledge and Information Systems. 2022 Jan; 64(1):35-74.

[6] Mateos P, Bellogín A. A systematic literature review of recent advances on context-aware recommender systems. Artificial Intelligence Review. 2025 Jan; 58 (1):1-53.

[7] Javed U, Shaukat K, Hameed IA, Iqbal F, Alam TM, Luo S. A review of content-based and context-based recommendation systems. International Journal of Emerging Technologies in Learning (IJET). 2021 Feb 12; 16(3):274-306.

[8] Afoudi Y, Lazaar M, Al Achhab M. Hybrid recommendation system combined content-based filtering and collaborative prediction using artificial neural network. Simulation Modelling Practice and Theory. 2021 Dec 1; 113:102375.

[9] Roy, D., Dutta, M. A systematic review and research perspective on recommender systems. J Big Data 9, 59 (2022). https://doi.org/10.1186/s40537-022-00592-5

[10] Chaudhari A, Alhussian H, Sarlan A, Raut R. A Hybrid Recommendation System: A Review. IEEE Access. 2024 Oct 14.

[11] Sivaramakrishnan N, Subramaniyaswamy V, Viloria A, Vijayakumar V, Senthilselvan N. A deep learning-based hybrid model for recommendation generation and ranking. Neural Computing and Applications. 2021 Sep; 33:10719-36.

[12] Arsytania IH, Setiawan EB, Kurniawan I. Movie recommender system with cascade hybrid filtering using convolutional neural network. Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI). 2024 Jun; 9(4):1262-74.

[13] Yang H, Gonçalves T. Field features: The impact in learning to rank approaches. Applied Soft Computing. 2023 May 1; 138:110183.

[14] Hu K, Li M, Song Z, Xu K, Xia Q, Sun N, Zhou P, Xia M. A review of research on reinforcement learning algorithms for multi-agents. Neurocomputing. 2024 Jun 18:128068.

[15] Ge, S., Dou, Z., Jiang, Z., Nie, J.-Y., & Wen, J.-R. (2019). Personalizing Search Results Using Hierarchical RNN with Query-aware Attention. ArXiv preprint arXiv: 1908.07600. https://arxiv.org/abs/1908.07600

[16] Keyhanipour AH. Graph-based comparative analysis of learning to rank datasets. International Journal of Data Science and Analytics. 2024 Mar; 17 (2):165-87.

[17] Lee J, Bernier-Colborne G, Maharaj T, Vajjala S. Methods, Applications, and Directions of Learning-to-Rank in NLP Research. InFindings of the Association for Computational Linguistics: NAACL 2024 2024 Jun (pp. 1900-1917).

[18] An GT, Choi WS, Park JY, Park JM, Lee KS. Search Re-ranking Through Weighted Deep Learning Model. The Transactions of the Korea Information Processing Society. 2024; 13(5):221-6.

[19] Chung J, Shin D, Hwang S, Park G. Horse race rank prediction using learning-to-rank approaches. The Korean Journal of Applied Statistics. 2024; 37(2):239-53.

[20] Semmelmann L, Resch O, Henni S, Weinhardt C. Privacy-preserving peak time forecasting with Learning to Rank XGBoost and extensive feature engineering. IET Smart Grid. 2024 Apr; 7(2):172-85.