Journal of Information & Optimization Sciences
ISSN 0252-2667 (Print), ISSN 2169-0103 (Online)
Vol. 46 (2025), No. 2, pp. 541–551
DOI : 10.47974/JIOS-1958

Optimizing learning to rank models with neural network-based feature selection techniques

Sushilkumar Chavhan *
R. C. Dharmik †
Department of Information Technology
Yeshwantrao Chavan College of Engineering
Nagpur 441110
Maharashtra
India

Abstract

Learning-to-rank (LTR) is a very important task in information retrieval. However, in today's era of exponentially growing data volumes available over the internet, acquiring high-quality data for feature selection poses a significant challenge in LTR. The emerging method of employing neural network-based feature selection in learning to rank utilizes machine learning methodologies to identify and choose relevant features, to enhanced efficiency and improve the ranking. This strategy overcome the constraints of conventional LTR methods and enhances the ranking performance across diverse applications. This research presents a new iteration of a sequential neural network model through the utilization of existing models, along with an analysis of its attributes. A distinctive approach to feature selection utilizing neural networks is outlined in this research for learning to rank, employing LambdaMART, a gradient boosting algorithm, to enhance the effectiveness and efficiency of ranking protocols on extensive datasets. The proposed method performance is better than the few existing state-of-arts algorithms. The model performance found as 0.711 NDCG, 0.61 P@K,0.611 on MSLR Web 30k and proposed model useful in many real time ranking applications.

Subject Classification: *68T05, 92B20.*

Keywords: *Learning to rank, Machine learning, Feature selection, Gradient boosting, Neural network.*

* *E-mail:* **schavhansushil@gmail.com (Corresponding Author)**
† *E-mail:* **raj_dharmik@yahoo.com**

© Taru Publications

1. Introduction

In past decade information is growing and it is overloaded, efficiently retrieving relevant documents from different repositories has become a significant challenge. Conventional techniques in information retrieval (IR) often lack precision and scalability, prompting the investigation of more advanced methodologies. Learning to Rank (LTR) is a sophisticated approach that emphasizes the arrangement of documents to ensure the most pertinent ones are displayed prominently in search outcomes. In this field, neural networks have gained prominence as effective instruments for capturing intricate, non-linear data relationships. This study introduces an innovative neural model designed for feature selection to optimize the learning-to-rank procedure, thereby improving the accuracy and efficiency of IR systems. Leaning to rank methods are categories into major three approaches. Pointwise approach, pairwise approach and listwise approach. The pointwise approach treats classification as a regression or classification problem on each document or quires. Score are evaluated and ranking was based on the scores. Majorly regression in which logistic regression or decision trees are popularly used. Burges et al. [8] laid the foundation for this method by using gradient descent to optimize the ranking functions. The pairwise method focuses on comparing pairs of documents to determine their relative rankings. The goal is to minimize the number of poorly ordered pairs. This approach is exemplified by Rank SVM [18] and RankNet [18], where the model is trained to predict the best document in a pair. The Listwise method takes into account the entire list of documents when learning to rank. These methods aim to directly optimize ranking metrics such as NDCG or MAP. The authors [9-10] introduced this approach and it is found to be effective in capturing the overall ranking structure.

The determination of capacity in LTR is important to progress the implementation of models by reducing dimensions, reducing excessive installation and improving interpretation. Feature selection strategies can be classified into the following categories: filter methods, wrapping methods and embedded methods. Filter Methods: These methods select characteristics based on their inherent properties without the association of learning algorithms. Common techniques include statistical tests, correlation coefficients, and mutual data. For example, TF-IDF [1] is a widely used text data filtering method. Wrapping Method: Wrapping Method evaluates the execution of a specific learning algorithm based on a subset of features. They include searching for the space of the functional

subsets and choosing the ones that produce the best performance. Recursive function elimination (RFE) and forward selection are examples of wrapper methods. Embedded Methods select functions in the model training process.

There are several challenges and arenas for prospective study in using neural feature selection for learning to rank. One of them is how to handle biases in user clicks like position bias, trust bias, and presentation bias which affect the performance of rankers [11]. Another one is about the size and complexity of neural architectures which restrict their application in large scale search systems due to latency and update time concerns [12]. Next research will focus on making feature selection more efficient by trading off classification performance against number of features and enhancing the diversity and convergence of the initial populations through novel strategies such as Elite Selection Strategy (ESS). Moreover, studying graph based approaches to model the relationships between resources and queries may increase the effectiveness of learning-to-rank systems under distributed information retrieval scenarios [13].

In this proposed approach neural network is used for feature selection with state of the art ranking algorithms. The components of presented solution are feature extraction and selection. In this method TF-IDF [1] is used to preprocess the input data to extract features. This step converts the raw text into numerical format for neural networks. Neural Network Architecture Following this model uses neural network architecture for ranking presented by authors [14]. The architecture has layers to capture feature interactions and relationships. Optimization and Training where Adam model used [15] to train the neural network to minimize ranking loss functions. This step also includes regularization to prevent overfitting and improve generalization. Evaluation and benchmarking where evaluate the model using IR metrics and compare with baselines and state of the art models using LETOR [16] as a benchmark to evaluate the model.

Some key contributions made by this study to the field of learning to rank include:

- By combining neural networks and feature selection mechanisms, our model bridges a critical gap in existing LTR techniques.
- The model suggested has been evaluated on benchmark datasets and shows better performance in ranking tasks generally.

- Our proposed model can be used for scaling purposes on other IR applications such as document retrieval, recommendation systems etc.

The neural model introduced here represents an important advance over extant studies in feature selection for learning to rank. It is designed to employ the power of deep learning frameworks while optimizing feature selection approaches, thus improving the efficiency, accuracy and scalability of modern IR systems. The following sections discuss its

Table 1

Overview of neural network model for learning to rank

Reference	Year	Contribution	Conclusion
[2]	2021	Introduced NeuRank, a neural network model for drug-target interaction prediction. Utilized deep learning to capture complex interactions.	Neural networks significantly outperform traditional methods in predicting drug-target interactions.
[3]	2019	Developed TF-Ranking, a scalable library supporting pointwise, pairwise, and listwise ranking methods.	Provides a robust and scalable platform for implementing various LTR models.
[4]	2020	Integrated BERT with TF-Ranking to enhance learning-to-rank models. Leveraged BERT's contextual understanding capabilities.	BERT's deep contextual embeddings significantly boost the performance of LTR models.
[5]	2018	Explored the application of deep neural networks for ranking tasks. Demonstrated improvements in ranking accuracy.	Deep neural networks offer substantial performance gains over traditional methods for ranking tasks.
[6]	2020	Revisited pairwise learning to rank using neural networks, providing theoretical analysis and practical performance insights.	Properly tuned neural networks can achieve superior performance in pairwise ranking tasks.
[7]	2019	Discussed the use of deep neural network regularization to prevent overfitting and improve model generalization.	Regularization is essential for maintaining robustness and accuracy of deep learning models in ranking.

methodological aspects, experimental design and findings, revealing an overall layout on how this model was developed and performed.

This table 1 provides a concise summary of neural-based models for learning to rank, highlighting their contributions and conclusions.

3. Problem Formulation

Given a set of training data $\{\{(x_i, y_i)\}_{i=1}^{N}$, where x_i represents the feature vector of the i-th document and y_i is its relevance score, the goal is to learn a ranking function f(x_i; θ) that accurately predicts the relevance scores (Table 2).

Table 2

Notation used in learning to rank

Notation	
D	The set of all documents or items to be ranked.
Q	The set of all queries.
x_i	Feature vector representing the i-th document.
y_i	Relevance score of the i-th document with respect to a given query.
f(x)	The ranking function or model that outputs a score for a given feature vector x.
θ	Parameters of the neural network model.

Feature Vector $x_i \in \mathbb{R}^d$ where is d is number of features. For feature section neural network is used having different layers, in convolution weights and bias is combine using following equation.

$$C(x_i) = Conv(x_i, W_c) + b_c \tag{1}$$

Dropout layer computation is done as where p is dropout layer which is assumed as standard value.

$$D(C(x_i)) = Dropout(C(x_i), p) \tag{2}$$

Flatten layer computation are as follows

$$F(D(C(x_i))) = Flatten(D(C(x_i)) \tag{3}$$

The single dense layer computations are as follows where W_1, W_2 and b_1, b_2 are weighs and bias of dense layer and σ is the relu action function.

$$h_1 = \sigma(W_1 \times F(D(C(x_i))) + b_1) \tag{4}$$

$$h_2 = \sigma(W_2 h_1 + b_1) \tag{5}$$

The output is computed as where W_0 and b_0 are the weigh and bias of the output layer.

$$f(x_i : \theta) = W_0 h_2 + b_0 \tag{6}$$

The above model is train using following lost function which is based on Ranknet

$$L(\theta) = \Sigma_{(i,j)\in P} \log(1 + exp(-(f(x_i;\theta)) - (f(x_j;\theta)))) \tag{7}$$

Where p is set of document pair

For proper selection of features a regularization term is added to loss function to deal with sparsity in each layer weights as,

$$L_T(\theta) = L(\theta) + \lambda \Sigma_{k=1}^{d} |W_{k,input}| \tag{8}$$

Where λ is regularization parameter and $W_{k,input}$ are weights in each of layer.

4. Experimental Setup

The experimental setup for evaluating a neural network-based feature selection model in a learning-to-rank (LTR) task involves several steps, from data preparation and model implementation to evaluation metrics and analysis.

4.1 *Datasets*

The datasets used for the work are standard learning to Rank datasets. The datasets are maintained by Microsoft [17]. The Datasets are shown in below table 3:

Table 3

Database used

Dataset	Benchmark	#Queries	Documents	#Relevance	#Features
MQ2007	LTEOR4.0	1692	70k	3	46
MQ2008	LTEOR4.0	784	15k	3	46
MSLR-WEB10K	Microsoft	10000	1200k	5	136
MSLR-WEB30K	Microsoft	31531	3775k	5	136

4.2 *Pre-processing*

Pre-processing is done by Normalize or standardizes feature values and using Alternating Least Squares. By using both the algorithms missing values are handle properly.

4.3 *Model architecture*

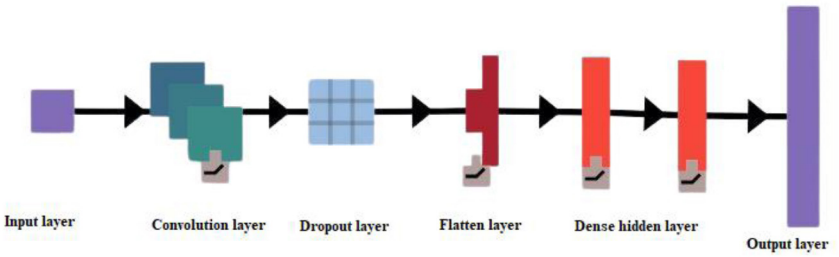


Figure 1

Architecture of Neural Network

The feature selection for learning to rank is done using neural network. By extracting higher-level feature from the input data and producing an output decision. Proposed model is sequential model, which means that each layer is connected to the previous layer in a sequential manner. Neural networks consist of Input layer of input corresponding to the feature vector followed by Convolutional layer to capture local feature patterns with Dropout layer to prevent overfitting, Flatten layer to convert the data to a single dimension followed by, Two dense hidden layers for complex feature interactions connected to output layer for relevance score. Detail diagram shown in fig1. Above neural network layers have a decreasing number of neurons in the subsequent layer. Convolutional layer has filter size of layer the convolutional 46/146, number of filters 32 and padding = 'same' and activation function is Relu. Input X with dimensions (batch_size, input_shape), Convolution operation done with activation function

$$A = \sigma(Z^l) \tag{9}$$

Where $$Z^l = W^l \times X + b^l \tag{10}$$

D is binary mask with probability p with initial value 1. It is applied in dropout layer. Element are arrange as per requirement in flatten layer and feature are selected. Dense layer is used proper classification for proper feature selection. Softmax function is used in this layer.

5. Result

This section provides the results of neural network and compares the various evaluation parameters as NDCG,P@K, MAP and Sparsity ratio. Model is trained by using all the all the mentioned datasets in table 1. Following table 4 shows the evaluations done after the feature selection by proposed neural model and train learning to rank by ranknet.

Table 4

Evolutionary Results

Metrics/ Dataset	MQ2007	MQ2008	MSLR-WEB10K	MSLR-WEB30K
NDCG@10	0.43	0.4	0.88	0.711
NDCG@3	0.389	0.48	0.86	0.814
P@3	0.74	0.557	0.992	0.751
P@10	0.7	0.623	0.98	0.61
MAP	0.38	0.4	0.66	0.611
Sparsity Ratio	0.81	0.85	0.296	0.4

The Table 4. Shows the comparison of the results of all the dataset for five folds. Value shown is the average of the entire fold. All the score indicate that the proposed model perform better than the benchmark score of LambdaMART. The sparsity ratio is also found to be better in case of the larger datasets whereas in the case of smaller datasets the results are lower than the expected results in the figure 2.

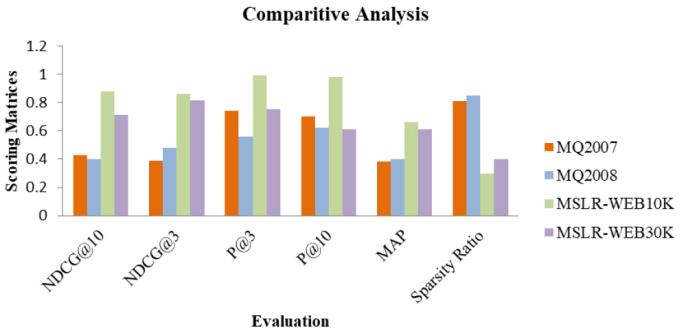


Figure 2

Comparative Analysis on all databases.

Above diagram shows the comparative analysis of each evaluation NDCG, P@K , map and sparsity raio. In MSLR database all the results are better as compare the other datasets. Proposed algorithm is tested with other neural based rank algorithms such as RankSVM, Ranknet, Listnet.

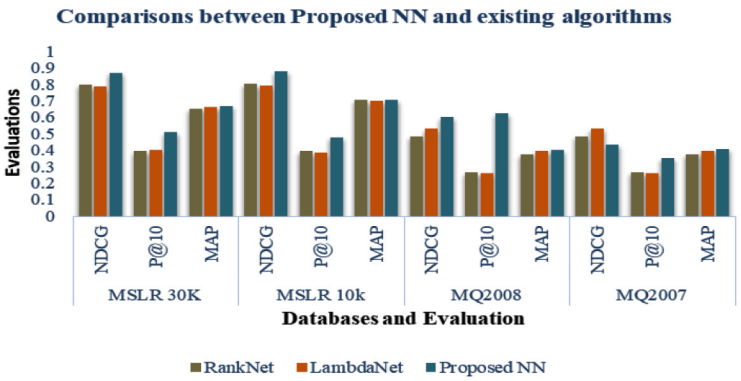


Figure 3

Comparisons with existing algorithms

In the figure 3 depicted that with all evaluation parameters it proposed model perform better with all data set. Listnet, RankSVM are popularly used model for leaning to rank. Where Listnet is neural network based algorithm and RankSVM is a machine learning algorithm for learning to rank. In both the case ∓0.5 increment is found. Sparsity Ratio of this system need to be handling so that it performs better with respect to time.

Conclusion

In this work development of neural network-based feature selection approach and performance analysis was done for learning to rank. The results demonstrate that thinning the network by sequencing layers and weights effectively selects the most relevant features from the input layer and removes inactive neurons from each layer. The model's performance was evaluated using benchmark datasets for learning to rank, showing that the compact network performed well across all benchmarks. Proposed model give the 0.711 NDCG, 0.61 P@K,0.611 map value for largest database MSLR Web30K. While selecting the relevant feature the sparsity raio is major issue to deal with. As a future direction used of presented model

with more dense layer to handle sparsity raio and apply for different applications where ranking is crucial task.

References

[1] S. Jabri, A. Dahbi, T. Gadi, and A. Bassir, "Ranking of Text Documents using TF-IDF Weighting and Association Rules mining," IEEE (2018).

[2] X. Wu, W. Zeng, F. Lin, and X. Zhou, "NeuRank: learning to rank with neural networks for drug-target interaction prediction," *BMC Bioinformatics* (2021).

[3] R. K. Pasumarthi, S. Bruch, X. Wang, C. Li, M. Bendersky, M. Najork, J. Pfeifer, N. Golbandi, R. Anil, and S. Wolf, "TF-Ranking: Scalable TensorFlow Library for Learning-to-Rank," *Proc. KDD '19: Applied Data Science Track Paper*, Anchorage, AK, USA, pp. 1–9 (2019).

[4] S. Han, X. Wang, M. Bendersky, and M. Najork, "Learning-To-Rank With BERT In TF-Ranking," (2020).

[5] C. Prakash and A. Sarkar, "Ranking with Deep Neural Networks," *2018 Fifth International Conference on Emerging Applications of Information Technology (EAIT)*, Kolkata, India, pp. 1–4 (2018).

[6] M. Köppel, A. Segner, M. Wagener, L. Pensel, A. Karwath, and S. Kramer, "Pairwise Learning to Rank by Neural Networks Revisited: Reconstruction, Theoretical Analysis and Practical Performance," *Lecture Notes in Computer Science (LNAI)*, vol. 11908, pp. 1–12 (2020).

[7] A. Rahangdale and S. Raut, "Deep Neural Network Regularization for Feature Selection in Learning-to-Rank," *IEEE Access*, vol. 7, pp. 53988–54006 (2019).

[8] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proc. 22nd International Conference on Machine Learning (ICML)*, pp. 89–96 (2005).

[9] S. Sharma, L. Raja, V. Bhatnagar, D. Sharma, S. N. Bhagirath, and R. C. Poonia, "Hybrid HOG-SVM encrypted face detection and recognition model," *J. Discrete Math. Sci. Cryptography,* vol. 25, no. 1, pp. 205–218 (2022).

[10] Z. Cao, T. Qin, T. Y. Liu, M. F. Tsai, and H. Li, "Learning to rank: From pairwise approach to listwise approach," (2007).

[11] L. Yu, Y. Wang, X. Sun, K. Bi, and J. Guo, "Feature-Enhanced Network with Hybrid Debiasing Strategies for Unbiased Learning to Rank," in *Proc. 16th ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 1–12 (2023).

[12] A. Purpura, K. Buchner, G. Silvello, and G. A. Susto, "Neural Feature Selection for Learning to Rank," in *Advances in Information Retrieval*, D. Hiemstra et al., Eds., Springer, pp. 1–10 (2021).

[13] U. Ergashev, E. Dragut, and W. Meng, "Learning To Rank Resources with GNN," in *Proc. ACM Web Conference 2023 (WWW '23)*, pp. 3247–3256 (2023).

[14] C. Prakash and A. Sarkar, "Ranking with Deep Neural Networks," *2018 Fifth International Conference on Emerging Applications of Information Technology (EAIT)*, Kolkata, India, pp. 1–4 (2018).

[15] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *International Conference on Learning Representations (ICLR)* (2014).

[16] T. Qin and T. Y. Liu, "Introducing LETOR 4.0 datasets," (2013).

[17] N. P. Sejdiu, B. Ristevski, and I. Jolevski, "Performance Comparison of Machine Learning Algorithms in Movie Recommender Systems," *2022 57th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST)*, Ohrid, North Macedonia, pp. 1–4 (2022).

[18] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proc. 22nd International Conference on Machine Learning (ICML)*, pp. 89–96 (2005).

Received November, 2024