Explanations for black-box rankers with Generalized Additive Models

Zhiheng Wang

Explanations for black-box rankers with Generalized Additive Models

by

Zhiheng Wang

to obtain the degree of Master of Science
at the Delft University of Technology,

Student number:	5476259	
Date:	May 27, 2024	
Thesis committee:	Prof. dr. Avishek. Anand,	TU Delft, chair
	Prof. dr. Soham. Chakraborty,	TU Delft, second examiner
	Ir. Lijun. Lyu,	TU Delft, daily supervisor

Abstract

Machine learning has revolutionized recommendation systems by employing ranking models for personalized item suggestions. Despite their effectiveness, learning-to-rank (LTR) models often operate as complex systems, making it difficult to discern the factors influencing their ranking decisions. This lack of transparency raises concerns about potential errors, biases, and ethical implications. As a result, interpretable LTR models have emerged as a solution to enhance transparency and mitigate these challenges.
Currently, the state-of-the-art intrinsically interpretable ranking model is led by generalized additive models. However, ranking GAMs have some limitations that affect their successful application in experiment environments, such as being computationally intensive and struggling to handle high-dimensional data. In contrast to these drawbacks, post-hoc methods can potentially provide more scalable and efficient solutions for real-time ranking.
In this study, we propose a post-hoc method for learning-to-rank tasks combined with the interpretable GAMs. The evaluation results tested with Kendall's τ value indicate that our model can effectively explain different types of black-box rankers.

Keyword:*Learning to rank, Interpretability in Ranking, Generalized Additive Models, Post-hoc Method, Kendall's τ value*

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Prof.Avishek Anand, and Lijun Lyu, for their continuous support, guidance, and encouragement throughout my research and thesis writing.

I am also thankful to the members of my thesis committee, Prof.Soham Chakraborty, for his valuable time in my thesis defense.

I am also thankful to my family and friends, for their support, understanding, and the stimulating discussions we have shared. Their assistance and moral support have been crucial during challenging times.

Contents

1 Introduction

Chapter 1 presents an introduction to learning to rank and the significance of building the interpretable ranking models. Based on this research background, the chapter then proposes a research objective and research questions. An outline of the whole thesis is also provided.

1.1. Background

Machine learning models are increasingly used in many areas that are capable of autonomous learning and improvement over time, without human intervention. One common application of machine learning is in recommendation systems by employing ranking models for personalized item suggestions in a relevant order(Nawrocka et al., 2018). Unlike the traditional machine learning models that focus on tasks such as classification, regression, etc, the primary objective of ranking models is to order a set of items (e.g., documents, products, web pages) based on their relevance or utility to a particular query or user(Guo et al., 2019).
Learning to Rank (LTR) models are usually trained by many parameters to achieve high accuracy, so they tend to be complex. The complexity of ranking models can sometimes undermine their effectiveness because it is difficult for humans to understand the reasons behind a particular order. So-called "black boxes" are often problematic because it is difficult to understand the factors that influence the model's output. This lack of transparency can lead to potential errors, biases, and even unethical behavior. Therefore, there is an urgent need to develop interpretable LTR models (Zytek et al., 2022).
Interpretability refers to the degree to which humans can understand the reasoning behind a model's decisions or outputs (Miller, 2018). According to this definition, models that humans more easily understand are considered more interpretable. Typically, explanations for the ranking models can be categorized into the development of intrinsically interpretable models and model-agnostic explanations. The former refers to either interpretability by design or access to all the model parameters, such as decision trees, additive models, and attention-based networks. On the other hand, model-agnostic explanations aim to generate post-hoc explanations for an existing black-box model without accessing its internal structure. Although different approaches in these two categories have been proposed within the context of ranking tasks, limited research has been conducted on the combination of an intrinsically interpretable model with model-agnostic explanations.

1.2. Problem Definition

Currently, the state-of-the-art intrinsically interpretable ranking model is led by generalized additive models (GAMs).In Zhuang et al. (2020), their research introduced how to extend GAMs to ranking models and proposed a new framework for this problem. This study presented a neural network-based notation for neural ranking GAMs by effectively utilizing list-level contextual features. The conducted experiments show that the ranking GAMs outperform traditional GAMs while maintaining interpretability. These findings make the ranking GAMs a standard approach for applications in LTR tasks, as they are interpretable models with inherently transparent and explainable structures.

However, ranking GAMs have some limitations that affect their successful application in experiment environments. First, they can be computationally intensive, limiting their scalability in experiment environments that require real-time ranking. Second, as the number of predictor variables increases, GAMs may struggle to handle high-dimensional data because of the consequent increase in the number of parameters. This may lead to overfitting and poor generalization of unseen data. These are also common drawbacks of intrinsically interpretable models. In contrast to these drawbacks, post-hoc methods can potentially provide more scalable and efficient solutions for real-time ranking.

Present studies make a definition of these two types of methods, especially for their merits and weaknesses. However, no study is dedicated to the integration of these two methods and provides a quantitative evaluation.

1.3. Research questions

Main research question:

This dissertation proposes a post-hoc method for learning-to-rank tasks combined with the interpretable GAMs. Specifically, the research question that will be addressed is:

“How are post-hoc methods designed to generate ranking explanations locally combined with GAMs?”

Research questions:

This main research question is divided into the following two sub-questions:

1. How to construct the neural ranking GAMs within the framework of post-hoc methods?
2. What metrics or criteria can be employed to evaluate the efficacy and interpretability of the expected approach in generating ranking explanations?

1.4. Outline

The remainder of this document is organized as follows.

In Chapter 2, the key concepts and previous studies related to the interpretability in machine learning are described, especially in the context of ranking tasks. We provide a clear classification of these interpretability methods.

In Chapter 3, the core of this research is discussed, that is, constructing neural ranking GAMs to explain the prediction of several black-box rankers. This section outlines the design and implementation of our research method.

Chapter 4 provides a comprehensive analysis of the experimental results, while Chapter 5 discusses the conclusions drawn from this study and future work directions.

2 Related Work

2.1. Interpretability methods

Interpretability in machine learning refers to the ability to understand and explain the decisions or predictions made by a model in a human-understandable manner. The original idea for model interpretability research arises from incompleteness in problem formalization (Doshi-Velez and Kim, 2017). It means that for certain tasks it needs to not only get the prediction result but also know how it comes to the result.
With the development of model interpretability methods, a variety of studies have created a taxonomy to classify them. However, most interpretability methods contain a variety of properties, which means the classification of these methods is not a simple binary categorization problem. Recently, Molnar (2022) proposed a taxonomy that is widely recognized to classify the existing model interpretability methods, as shown in Figure 2.1.

- **Intrinsic Interpretability:** Intrinsic interpretability refers to machine learning models that are considered interpretable due to their simple structure, such as short decision trees or sparse linear models. In other words, an intrinsically interpretable model is one that naturally produces human-understandable explanations for its predictions. Generalized additive model(GAM) is one of the typical examples in this category:

  1. **GAM:** In (Hastie and Tibshirani, 1986), Generalized additive models(GAMs) are a class of statistical models that generalize linear regression models by allowing non-linear relationships between the predictors and the response variable. GAMs relax the restriction of the generalized linear models that the relationship must be a simple weighted sum and instead assume that the outcome can be modeled by a sum of outputs of multiple sub-models, where each sub-model only takes one feature as input. The motivation for applying GAMs is that each sub-model reflects the contribution of each input feature to the final output results, which makes them more interpretable than complicated models such as deep neural networks.
     In the context of ranking, GAMs can be adapted to model the relationship between features and ranking scores. One prominent example of interpretable ranking with the generalized additive models is proposed by Zhuang et al. (2020). In their work, Zhuang et al. (2020) employed standalone neural networks to instantiate sub-models in GAMs for each individual feature. This framework utilized the flexibility of neural networks for the ranking task while maintaining the intelligibility of traditional GAMs. To be specific, based on a user query, the local search system computes

a set of item-level features x_js for each item. Each sub-model only takes a single item-level feature x_j as input and outputs a corresponding sub-score $f_j(x_j)$. As for the ranking task, the context features should also be taken into account. The context features are to derive importance weights for item-level sub-scores before they are summed up as the final ranking score. This method ensures not only the interpretability of the item-level feature attribution but also the contribution of the context feature.

- **Post-hoc Interpretability:** Post-hoc interpretability refers to the application of interpretation methods after model training, which is also the method researched in this paper. Compared with intrinsic interpretability which involves designing models to be inherently transparent, post-hoc interpretability methods are applied to existing models, regardless of their complexity. Some common post-hoc interpretability methods include:

  1. **LIME:** As proposed by (Ribeiro et al., 2016), local interpretable model-agnostic explanations(LIME) is a machine learning technique designed to provide interpretable explanations for individual predictions made by complex machine learning models. The main idea is to train a simple linear model in a new interpretable feature space. The simple model is trained on a new dataset of perturbed samples paired with their corresponding prediction of the given model for a selected instance. After fitting the simple linear model, LIME provides feature importance scores, indicating which features had the most significant impact on the model's prediction for the specific instance.
     To illustrate LIME more specifically, given a black box classifier $f : x \rightarrow y$, where the input is s an n-dimensional feature domain $\mathbb{R}^n$, and the output is the posterior probability of a certain class in the classification task. In the local area around x, the prediction problem is simplified and the computation of f is assumed to be linear. A linear surrogate model $g \in G$ is defined to approximate the prediction result of f, where G is a set of potentially interpretable models. Since the interpretable model $g \in G$ can be readily presented to the user with visual or textual artifacts, the decision of f is explained by the interpretation of g.
     In order to approximate f, LIME perturbs the features of the instance x and generates a local data point dataset $z \in Z$. $\pi_x(z)$ is defined as a measure to approximate the extent of the local points around the instance x, which is a weight based on the distance to x. The points closer to the original instance x are given more importance in the surrogate approximation. The main objective of LIME is to train a simple model g that minimizes $L(f, g, \pi_x)$ which is a measure of how far g is in approximating f in the locality defined by π_x. The loss function $L(f, g, \pi_x)$ is the dissonance between the prediction of g and the label defined by f.

  2. **DeepLIFT:** Deep Learning Important Features is a gradient-based method for decomposing the output prediction of a neural network on a specific input by backpropagating the contributions of all neurons in the network to every feature of the input (Shrikumar et al., 2017). Formally, DeepLIFT attributes to each input x_i a value $C_{\Delta x_i \Delta o}$ that represents the effect of that input being set to a reference value as opposed to its original value. $C_{\Delta x_i \Delta o}$ can be non-zero even when $\frac{\vartheta o}{\vartheta x_i}$ equals to zero. That means for DeepLIFT, the fundamental limitation of gradients is solved since a neuron can signal meaningful information even in the regime where its gradient is zero. The reference value would be chosen based on domain-specific knowledge and usually represents a typical uninformative background value for the feature.

3. **SHAP:** Shapley Additive Explanations(SHAP)(Lundberg and Lee, 2017) is a game theoretic approach to explain the output of any machine learning model. It's based on cooperative game theory and calculates the contribution of each feature to the prediction made by the model. The aim is to assign each feature an "importance" score, indicating how much that feature contributes to the final prediction. In cooperative game theory, the "importance" score is called the Shapley value, which is a way to fairly distribute a total payout among a group of individuals based on their marginal contributions.
4. **Global surrogate Model:** The global surrogate model is used to approximate a "black-box" machine learning model with a simple and interpretable model. The purpose of it is to provide insights into how the complex model makes predictions by creating a transparent and understandable representation that mimics its behavior to some extent. For example, Singh and Anand (2020) proposed a framework to approximate a complex ranker by using a simple ranking model in the term space.

- **Global Interpretability:** Global model interpretability refers to the process of understanding and explaining the behavior and decision-making process of an entire machine learning model across its entire input space. It aims to provide insights into how the model as a whole operates, what features it relies on, and how those features contribute to its overall performance. Examples include Partial Dependence Plots (PDP) and Individual Conditional Expectation (ICE) plots.

- **Local Interpretability:** Local interpretability refers to the understanding and explanation of individual predictions made by a machine learning model. Unlike global interpretability, local interpretability focuses on explaining why a particular instance was assigned a specific prediction or classification. LIME is one of the most popular methods in this category.

- **Model-specific Interpretability:** Model-specific interpretability refers to the application of interpretability techniques that are used for a specific type of machine learning model. For example, visualizing the support vectors and decision boundaries that determine the classification is only used for support vector machine(SVM).

- **Model-agnostic Interpretability:** Model-agnostic interpretability refers to the application of interpretability techniques that can be used on any machine learning model and are applied after the model has been trained. Model-agnostic methods are especially good for dealing with complex models. SHAP, LIME, and the surrogate model(Singh and Anand, 2020) we mentioned before are all in this category.

2.2. Interpretability methods in Ranking

2.2.1. Learning to rank

Learning to rank is a supervised task to construct a ranking model using training data, such that the model can order a set of items given a query according to their degrees of relevance, preference, or importance (Liu et al., 2009). The main goal of learning to rank problems is to optimize the ranking results in order to maximize user satisfaction
Typically, a typical training set contains n query instances $q_i(i = 1, ..., n)$, each of them associated with documents represented by feature vectors $x^{(i)} = \{x_j{}^{(i)}\}_{j=1}^{m(i)}$, where $m(i)$ is the number of documents associated with query q_i and the corresponding ranked list y_i. From there, a particular learning algorithm is employed to train the ranking model to generate predictions that align with the actual ground truth labels as accurately as possible by using a loss

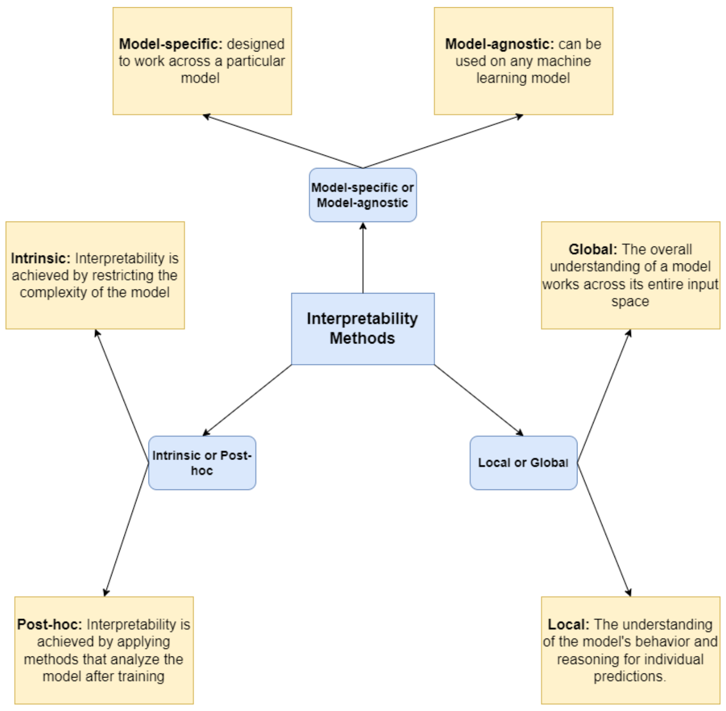

Model-specific: designed to work across a particular model
Model-agnostic: can be used on any machine learning model
Model-specific or Model-agnostic
Intrinsic: Interpretability is achieved by restricting the complexity of the model
Interpretability Methods
Global: The overall understanding of a model works across its entire input space
Intrinsic or Post-hoc
Local or Global
Post-hoc: Interpretability is achieved by applying methods that analyze the model after training
Local: The understanding of the model's behavior and reasoning for individual predictions.

Figure 2.1: The taxonomy of interpretability methods

function. In the test phase, the model previously trained is utilized to rank the documents for a new query based on their relevance to the query, and the ranking result is then returned to users.

2.2.2. Interpretability in ranking

As been exposed previously, interpretability in machine learning has been studied extensively in classical machine learning. However, the success of interpretability in ranking models is limited, because most of the previous works of interpretability in the ranking are based on the extensions to the existing pointwise(why is doc_i relevant ?) or pairwise explanations (why is doc_i ranked higher than doc_j?). For example, methods based on LIME framework generate terms as explanations for a black-cox ranker. DeepLIFT we mentioned in the previous section is one of the gradient-based methods to interpret the relevance score as a document to a given query. However, the ranking result is an entire list, which means it is not sufficient to give an explanation only based pointwise or pairwise. In this work, we instead focus on explaining the entire ranking list and provide the listwise explanation.

Typically, explanations for the ranking models can be categorized into the development of intrinsically interpretable models and model-agonistic explanations. The former refers to either interpretability by design or access to all the model parameters. In interpretability by design, more ”interpretable” models such as decision trees, rules(Letham et al., 2015), additive models(Zhuang et al., 2020), and attention-based networks are utilized. Letham et al. (2015) proposed a model called the Bayesian Rule Lists(BRL), which provides a new type of balance between accuracy, interpretability, and computation. Zhuang et al. (2020) provide a framework for intrinsically interpretable learning-to-rank by introducing generalized additive model(GAM) into ranking tasks.

On the other hand, model-agnostic explanations aim to generate post-hoc explanations for an existing black-box model without accessing its internal structure. As for the application in the ranking tasks, a few recent studies (Lyu and Anand, 2023; Singh and Anand, 2018a,b,

2020) focus on the model-agnostic method. In (Singh and Anand, 2018b), the authors tried to approximate a giving ranking model with a subset of interpretable features. Singh and Anand (2018a) proposed the EXplainable Search(EXS) system aimed to help users understand the trained retrieval model based on the LIME framework. They also attempted to approximate the output of complex rankers on a local view with a simple ranking model based on query expansion(Singh and Anand, 2020). Lyu and Anand (2023) challenged the assumption in (Singh and Anand, 2020) that a single relevance factor is adequate to explain an entire ranking and proposed the MULTIPLEX which combined multiple simple rankers instead.

2.3. Bi-Encoder vs Cross-Encoder

In the rapidly evolving field of Natural Language Processing (NLP), the ability to represent sentences accurately and efficiently is becoming critical. As information retrieval techniques advance, models like BERT have revolutionized our understanding in the world of sentence embedding. Among all the available models and techniques, two in particular have stood out recently: the Crossencoder and the Bi-encoder. In this section, we will give a comprehensive explanation of these two approaches, comparing their architectures, advantages, and weaknesses. The architectures of the Bi-Encoder and Cross-Encoder are shown in Fig2.2

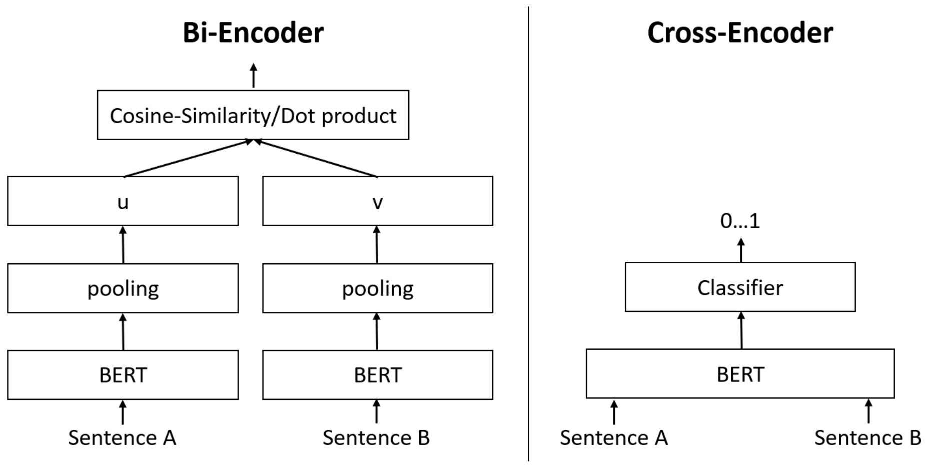


Figure 2.2: The architectures of Bi-Encoder and Cross-Encoder

2.3.1. Bi-Encoder

Bi-Encoders produce for a given sentence a sentence embedding. Sentence A and Sentence B are passed to a BERT separately, which results in the sentence embeddings u and v. For the ranking tasks, one sentence is a given query, and the other is one document in the corpus. These sentence embeddings can then be compared using cosine similarity or dot-product.

- **Advantages:**
  1. Speed: One of the advantages of Bi-Encoders lies in their computational efficiency. By separately encoding queries and corpus, they exhibit markedly reduced prediction times, particularly suited for real-time applications and large-scale tasks.
  2. Scalability: Bi-Encoders demonstrate remarkable scalability, adeptly managing extensive datasets by efficiently indexing encoded candidates and swiftly comparing these representations.

- **Challenges:**
  1. Accuracy Trade-off: Despite their computational efficiency, Bi-Encoders may occasionally exhibit a compromise in accuracy compared to Cross-Encoders, particularly in tasks requiring nuanced contextual comprehension.
  2. Representation Limitations: The separate encoding of queries and corpus in Bi-Encoders may result in instances where certain contextual subtleties are overlooked.

2.3.2. Cross-Encoder

In contrast, for a Cross-Encoder, both sentences are simultaneously fed into the Transformer network. Subsequently, the network generates an output value ranging between 0 and 1, indicating the similarity of the input sentence pair.

- **Advantages:**
  1. Accuracy: The comprehensive utilization of the full self-attention mechanism often positions Cross-Encoders to achieve superior accuracy, particularly in tasks necessitating profound contextual comprehension.
  2. Detailed Representations: Through the recalibration of encodings for each input-label pair, Cross-Encoders excel in generating intricate and contextually enriched sentence representations.

- **Challenges:**
  1. Computational Intensity: The requirement to recompute encodings for every input-label pair leads to Cross-Encoders being computationally intensive, particularly when dealing with large datasets.
  2. Slower Test Times: Given their intricate encoding procedure, Cross-Encoders may exhibit slower testing times, posing challenges for real-time applications.

2.4. Our contribution

Although different approaches that use local model-agnostic interpretability explanations have been proposed within the context of ranking tasks, limited research has been conducted to assess their capabilities combined with GAMs. In previous papers, local model-agnostic interpretability techniques produce explanations for a single decision by linearly approximating the local behavior of a complex blackbox model. However, a linear model is not sufficient to explain the non-linear relationship in deep neural networks. The thesis proposes a framework to approximate the ranking list from two types of black-box models (One is Bi-Encoder, the other is CrossEncoder)by replacing the linear model in the LIME framework with GAMs. The motivation for this method is to approximate the non-linear relationship in neural networks while maintaining the interpretability in the LIME framework.

3 Research Methodology

In Chapter 3, an overview of the research methodology is provided, including the fundamental framework of the experimental approach and a detailed description of its implementation. Detailed information on various aspects of the research methodology includes the datasets we used, the selected black-box rankers and tools, as well as the design of the neural ranking GAMs.

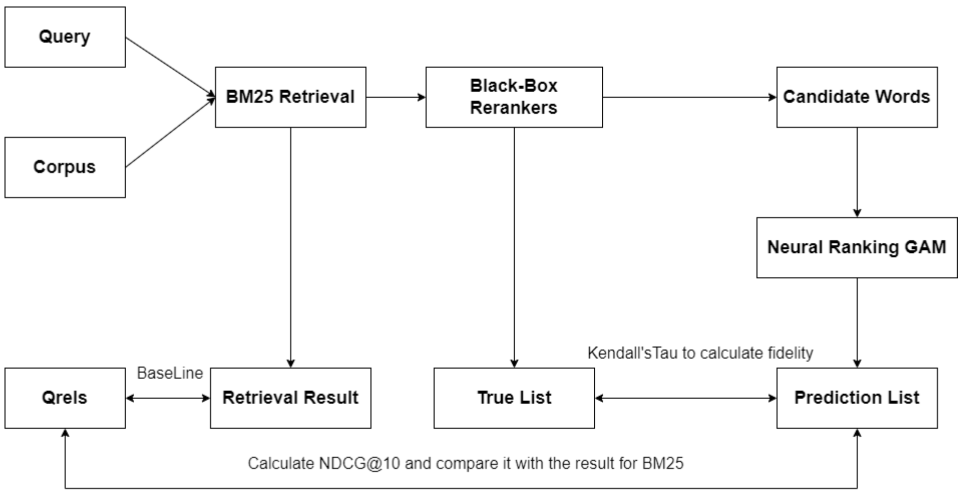


Figure 3.1: The overview of the research method

3.1. Methods

The research methodology outlined in this section is shown in Fig 3.1, integrating elements of retrieval, reranking, and evaluation techniques. For a given query in one dataset and its corresponding corpus, the retrieval phase is initiated utilizing the BM25 method, a widely adopted technique for information retrieval tasks. This process aims to retrieve all relevant documents to the specific query. Subsequently, we picked out a subset of the retrieved documents based on the task-specific criteria for reranking using a black-box ranker. From the reranked documents, we selected keywords to serve as inputs for the proposed neural ranking GAMs. These keywords are selected based on their significance in representing the content and context of the documents. The number of extracted keywords corresponds to the number of sub-neural networks within the GAMs.

The ground truth labels for training are derived from the ranking produced by the black-box ranker, providing a benchmark for assessing the performance of the neural ranking GAMs. During the testing phase, the models' fidelity is evaluated using Kendall's tau value, a metric commonly employed to measure the correlation between the predicted rankings generated by the neural ranking GAMs and the ground truth rankings.
In addition to fidelity assessment, the performance of the proposed models is further evaluated using the Normalized Discounted Cumulative Gain (NDCG) metric. This metric quantifies the effectiveness of the ranking algorithms in prioritizing relevant documents, with higher NDCG values indicating superior performance. A baseline comparison is conducted against the BM25 method for assessing the relative efficacy of the neural ranking GAMs.

3.2. Methods Application

3.2.1. Datasets

This study uses two datasets: SciFact and FiQA-2018. Both datasets have been established as widely accepted benchmarks in the fields of machine learning and information retrieval. Table 4.1 summarizes these datasets.

Datasets	SciFact	FiQA-2018
#Documents	5183	57000
#Queries	1407	6648
Avg#Doc per Query	1.1	2.6
#Train\Dev\Test	80%\0%\20%	80%\10%\10%

Table 3.1: Datasets Description

SciFact: The SciFact dataset is a comprehensive collection of factual claims extracted from scientific articles, accompanied by annotations denoting their veracity. This dataset serves as a valuable resource for training and evaluating fact-checking models and natural language understanding systems designed to retrieve and reason over corpus containing specialized domain knowledge. The SciFact dataset(Wadden et al., 2020) contains $1.4k$ expert-written scientific claims paired with evidence-containing abstracts annotated with labels and rationales.

FiQA-2018: The FiQA-2018 dataset(Boteva et al., 2016) is a Question-Answering English retrieval dataset in the financial information retrieval domain. The financial domain, characterized by its reliance on deciphering multiple unstructured and structured data sources and its imperative for swift and comprehensive decision-making, is increasingly becoming a principal area for the application of NLP, Web Mining, and Information Retrieval (IR) techniques. This dataset contained 6648 queries and 57k documents, designed to explore the state-of-the-art aspect-based sentiment analysis and opinion-based Question Answering for the financial domain.

3.2.2. BlackBox Ranking Models

The decision to adopt both Approximate nearest neighbor Negative Contrastive Learning(ANCE) and MS-MARCO-Electra-base as our black-box ranking models in this paper stems from their respective strengths and suitability for different aspects of information retrieval tasks. ANCE within a bi-encoder architecture excels in efficient similarity computation, making it ideal for rapid retrieval of relevant documents in large-scale systems. Conversely, MS-MARCO-Electra-base within a cross-encoder architecture offers superior contextual understanding, enhancing the model's ability to accurately rank passages based on query relevance. By selecting black-box ranking models with different structures, we aim to investigate whether this will affect the interpretability of the GAMs we designed. Additionally, it is worth noting that ANCE outputs document-query relevance scores using dot product, while MS-MARCO-Electra-base uses cosine similarity. This difference may affect our choice of loss functions, which we will explain in detail later.

- **ANCE**: ANCE, introduced by (Xiong et al., 2020), is a groundbreaking approach designed to enhance confidence estimation in approximate nearest neighbor search within large-scale retrieval systems. At its core, ANCE employs a bi-encoder architecture, a distinctive feature that sets it apart in the realm of information retrieval models. This architecture involves encoding both queries and documents separately, enabling efficient similarity computation for retrieval tasks. By leveraging negative contrastive learning techniques, ANCE significantly improves the effectiveness and scalability of retrieval systems by accurately estimating confidence scores for candidate documents. This model is particularly well-suited for large datasets in various information retrieval applications.

- **MS-MARCO-Electra-base**: MS-MARCO-Electra-base is a state-of-the-art pre-trained language representation model developed by UKP Lab, building upon the foundational framework of the Electra model proposed by (Clark et al., 2020). This model is specifically trained on the MS MARCO dataset, a large-scale dataset designed for a myriad of natural language understanding tasks, including passage ranking and question-answering. MS-MARCO-Electra-base utilized a cross-encoder architecture. Unlike bi-encoder architectures, cross-encoder architectures encode input sentences jointly, allowing them to capture intricate contextual dependencies and semantic relationships more effectively. Through fine-tuning on the MS MARCO dataset, MS-MARCO-Electra-base demonstrates exceptional proficiency in accurately ranking passages based on query relevance. In the subsequent sections, we will refer to this model as ”ME” for brevity.

3.2.3. Experimental Setup

The research question in Chapter 1 is divided into three subtasks, which encompass a comprehensive evaluation of Generalized Additive Models (GAMs) across diverse retrieval scenarios, each designed to address specific aspects of the model performance, effectiveness, and scalability. Through these experiments and analysis, we aim to gain a profound understanding of GAMs' capabilities and limitations, thereby informing their practical applicability in real-world information retrieval tasks.

- **Task1: GAMs on one query**
  Task 1 sets the foundation for evaluating GAMs' effectiveness by constructing the neural ranking GAMS based on the top-20 documents retrieved using the BM25 method for one given query. This task aims to assess the model's ability to replicate the ranking results of the original black-box ranker on these most relevant documents. By comparing the ranking results and calculating Kendall's Tau value, we can quantify the degree of correlation between the GAMs' rankings and those generated by the black-box ranker, providing the evaluation for the model's performance and fidelity.

- **Task 2: GAMs on the SciFact dataset**
  Expanding on the evaluation scope, Task 2 aims to conduct a comprehensive assessment of GAMs using the whole SciFact dataset. With a diverse array of queries encompassing various topics, this task provides a holistic evaluation of the GAMs' performance

across a wide spectrum of ranking scenarios. The retrieval process entails retrieving the top-10 documents for each query in the SciFact dataset utilizing the BM25 method. With a total of 1109 queries available, the SciFact dataset is partitioned into two sets for training (887 queries) and testing (222 queries). Similar to Task 1, Kendall's Tau value is computed for each query in the test set, assessing the degree of correlation between the ranking lists generated by the GAMs and those produced by the original black-box ranker. By analyzing the ranking results across all queries in the dataset, we can evaluate the GAMs' overall performance and effectiveness in replicating the ranking outcomes of the black-box ranker across the whole dataset, thereby informing their suitability for practical applications in real-world scenarios.

- **Task 3: Optimization and evaluations of the GAMs**
  Due to the extended training time required for Task 2, Task 3 focuses on assessing the scalability of GAMs through optimization strategies. With a focus on optimizing training time and computational resources, we optimize the research method in Task 2 to expedite the training process while maintaining optimal performance. In this task, a subset of 150 queries is selected from the datasets to mitigate computational demands. For each selected query, the top-100 documents are retrieved using the BM25 method, and every tenth document is selected, resulting in ten documents per query. The GAMs are then constructed and trained on these selected documents, with 120 queries used for training and the remaining 30 queries for testing. Kendall's Tau value is calculated for each query in the test set to evaluate the scalability and performance of GAMs with optimized training strategies. This task aims to assess the feasibility of using GAMs in scenarios where computational resources are limited and training time is a critical factor.

3.3. The Neural Ranking GAMs

As we mentioned in Chapter 2, Zhuang et al. (2020) proposed an interpretable ranking model based on Generalized Additive Models (GAMs). In their framework, standalone neural networks were used to instantiate sub-models within GAMs for individual features. This approach combined the flexibility of neural networks with the interpretability of traditional GAMs. Specifically, the model computed item-level features for each item based on a user query. Each sub-model took a single item-level feature as input and produced a corresponding sub-score. Drawing inspiration from this research, we designed our interpretable ranking model for this paper. The process of designing the GAMs consists of four main parts: candidate term selection, choice of loss function, network architecture design and hyperparameter settings.

3.3.1. Candidate Term Selection

Candidate term selection involves identifying keywords that represent the documents corresponding to each query for ranking, and using these keywords to construct a vocabulary. Each word in this vocabulary serves as an input feature for our GAMS model. For Task 1, we begin by pre-selecting an initial set of terms from the top 20 documents of the ranking list for a given query. This pre-selection is conducted after removing common and generic terms through TF-IDF filtering. In our experiments, we initially chose 400 terms. However, not all of these terms contribute meaningfully to the explanation, and many may be false positives. To mitigate this issue, we employed a systematic perturbation method. This involved iteratively removing and adding terms to the documents, and evaluating the impact of these changes. Through this process, we refined our selection and retained the top 200 terms, which served as our candidate term set.

For Task 2 and Task 3, we utilized a similar methodology as described in Task 1 to select candidate terms for each query. The candidate terms were again chosen through a combination of TF-IDF filtering and systematic perturbation. After selecting candidate terms for each individual query, we aggregated all the candidate terms from each query, including those from both the test set and the training set. This aggregation resulted in a comprehensive vocabulary that encapsulates a broad range of relevant terms across all queries.

3.3.2. Choice of Loss Function

In our research, we have employed two distinct list-wise loss functions to optimize the ranking performance of our model: Mean Squared Error (MSE) loss and ListNet loss. We will compare and evaluate the differences in results by combining these two loss functions with the two types of black-box models as described before.

Mean Squared Error (MSE) Loss:

Mean Squared Error (MSE) loss is a widely-used loss function, particularly in regression tasks. It measures the average of the squares of the errors, that is the difference between the predicted values and the actual values. Formally, MSE is defined as:

$$MSE = \frac{1}{n}\Sigma_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.1}$$

where y_i represents the actual value, $\hat{y}_i$ represents the predicted value, and n is the number of relevant documents.

In the context of document ranking, MSE loss evaluates the discrepancy between the predicted relevance scores of the documents and the actual relevance scores. By minimizing the MSE loss, the model is trained to produce relevance scores that are as close as possible to the true relevance scores. This approach is advantageous because it penalizes larger errors more heavily, thereby encouraging the model to make accurate predictions. The use of MSE loss in ranking ensures that the predicted relevance scores maintain a close alignment with the actual relevance scores, which is crucial for effective ranking.

ListNet Loss:

ListNet loss, on the other hand, is specifically designed for ranking tasks. It is based on the ListNet learning to rank algorithm, which focuses on directly optimizing the order of the items rather than their absolute scores. ListNet uses a probability distribution over permutations to define the loss function. The key idea is to minimize the distance between the probability distributions of the predicted and actual permutations of the documents.

ListNet loss is defined using the top-one probability, which considers the probability of each document being the top-ranked document. This probability is computed using a softmax function over the predicted scores. Formally, the top-one probability for a document d_i is given by:

$$P(d_i) = \frac{e^{\hat{y}^i}}{\Sigma_j e^{\hat{y}^j}} \tag{3.2}$$

where $\hat{y}^i$ is the predicted score for document d_i.

The ListNet loss function then measures the cross-entropy between the predicted top-one probabilities and the actual top-one probabilities derived from the ground truth rankings. This is formally expressed as:

$$ListNetLoss = -\Sigma_i P(d_i) log \hat{P}(d_i) \tag{3.3}$$

where $P(d_i)$ is the actual probability distribution and $\hat{(P)}(d_i)$ is the predicted probability distribution.

By minimizing the ListNet loss, the model is trained to produce a ranking order that closely matches the true ranking order. This approach is particularly beneficial for ranking tasks because it directly optimizes the order of the documents, which is the primary objective in ranking scenarios.

Comparison and Use in Our Model: Both MSE loss and ListNet loss have their distinct advantages. MSE loss is straightforward and ensures that the predicted scores are numerically close to the actual scores. This is useful for tasks where the precise value of the scores is important. However, in ranking tasks, the order of the documents is often more important than the exact scores. This is where ListNet loss excels, as it directly optimizes the ranking order by considering the permutations of the documents.

In our experiments, since the ANCE model outputs document scores based on dot product, the values of the ground truth score are typically large and not within the range of 0 to 1. Using MSE loss may result in excessively large loss values, causing the model to converge slowly. We utilize both MSE and ListNet loss function to compare their effectiveness.

3.3.3. Network Architecture Design

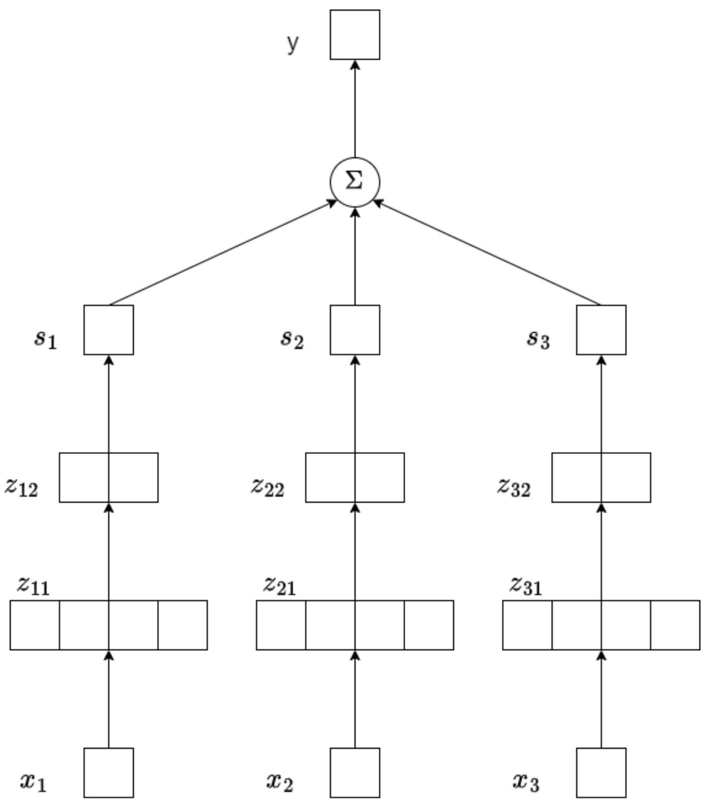


Figure 3.2: A graphical illustration of a neural ranking GAM

This section primarily describes the design and construction of the nueral ranking GAM structure used in this study for document ranking.For a given query, our approach involves building a scoring function that predicts the ranking score y for each corresponding document in the ranking list using a neural ranking GAM. Each document can be represented by several words from the selected vocabulary, with each candidate term in a document taken as a feature x_i. To achieve this, we designed a standalone neural network for each feature x_i. Each of these neural networks outputs a single "subscore" $s_i \in R$. Consequently, we have n separate neural networks in total, where n is the number of features. We adopted pre-trained GloVe word vectors for word embedding in our task, which transform each word into a 100-dimensional vector. These embeddings provide a rich representation of the semantic information of the words.

For each feature, we constructed a feed-forward network with Lhidden layers, as depicted in Figure 3.2. Each sub-network processes its respective feature to output a subscore. The final predicted score y for a document is obtained by summing all the subscores from the individual feature networks:

$$y = \Sigma_{i=1}^{n} s_i \tag{3.4}$$

This additive structure allows each feature to contribute independently to the final score, aligning with the principles of GAMs, where the effect of each feature is modeled separately and then combined. During training, we stacked the predicted scores for all documents to construct a predicted ranking list for each query. The predicted ranking list is then compared against the ranking results produced by the original black-box ranker. The loss is computed based on this comparison, guiding the optimization of the neural networks

3.3.4. Hyperparameter Settings for the Neural Ranking GAM

Task	Task1		Task2	Task3			
#Dataset	SciFact	FiQA-2018	SciFact	SciFact		FiQA-2018	
#Black-box ranker	ANCE	ME	ANCE	ANCE	ME	ANCE	ME
#Batch Size	1		8	8			
#Number of steps for train	100	100	3	30		30	
#Number of sub-networks	200	200	4k	3.3k		3k	
#Size for hidden layers	50,30		50,30	50,30			
#Learning rate	0.001		0.001	0.001			
#Loss	ListNet	MSE	ListNet	ListNet	MSE	MSE	ListNet

Table 3.2: Hyperparameter values for the neural ranking GAM

Table 4.2 provides an overview of the hyperparameter settings used for the neural ranking GAM across three different tasks. These tasks involve different datasets, black-box rankers, batch sizes, training steps, number of sub-networks, hidden layer sizes, learning rates, and loss functions.
For Task 1, we use the SciFact and FiQA-2018 datasets, with the ANCE model as the black-box ranker for SciFact and the ME model for FiQA-2018. Both datasets have a batch size of 1, and the training involved 100 steps. Each task uses 200 sub-networks, indicating the selection of 200 distinct features. The neural networks for each feature have hidden layers of sizes 50 and 30, and the learning rate is set at 0.001. The loss functions differed: ListNet loss is used for SciFact, while MSE loss is used for FiQA-2018.
In Task 2, the SciFact dataset is evaluated with the ANCE black-box ranker. This task has a larger batch size of 8, suggesting the need for efficient batch processing due to the dataset's size. The training is preliminary, with only 3 steps, and involves 4000 sub-networks, reflecting a comprehensive feature set. The hidden layer sizes and learning rate remain the same as in Task 1. ListNet loss is used to optimize the ranking performance.
Task 3 involves a comparative evaluation using both SciFact and FiQA-2018 datasets, with both ANCE and ME rankers. The batch size is consistently set at 8, ensuring efficient training. Each configuration undergoes 30 training steps, indicating a more thorough training phase. The number of sub-networks varies slightly, with 3300 for SciFact and 3000 for FiQA-2018. The hidden layer sizes and learning rate are consistent across all configurations. The loss functions vary to explore their impacts: ListNet loss is used with ANCE for SciFact and with ME for FiQA-2018, while MSE loss is used with ME for SciFact and with ANCE for FiQA-2018.

3.4. Evaluation Metrics

In this study, we employed two primary evaluation metrics to assess the performance of our neural ranking Generalized Additive Model (GAM): Normalized Discounted Cumulative Gain (NDCG) and Kendall's τ value.

3.4.1. Normalized Discounted Cumulative Gain (NDCG)

NDCG is a widely used metric in information retrieval that measures the effectiveness of a ranking system based on the positions of relevant documents in the result list. The goal of NDCG is to provide a more nuanced evaluation than traditional metrics like precision and recall by considering the rank positions of the relevant documents. The relevance scores are typically discounted logarithmically, emphasizing the importance of higher-ranked documents:

$$DCG_p = \Sigma_{i=1}^{p} \frac{2^{rel_i} - 1}{log_2(i+1)} \tag{3.5}$$

Where rel_i is the relevance score of the document at position i, and p is the number of top results considered. The DCG is then normalized by the Ideal DCG (IDCG), which is the DCG of the ideal ranking where the most relevant documents are ranked at the top:

$$NDCG_p = \frac{DCG_p}{IDCG_p} \tag{3.6}$$

NDCG values range from 0 to 1, with 1 indicating a perfect ranking where the most relevant documents are positioned at the top. This metric is particularly useful for evaluating ranking models because it not only considers the relevance of the retrieved documents but also their order, thus providing a more comprehensive measure of the ranking quality.

3.4.2. Kendall's Tau Value

Kendall's τ value is a rank correlation coefficient that measures the ordinal association between two ranked lists. It evaluates the similarity between the predicted ranking and the ground truth ranking by counting the number of concordant and discordant pairs. A pair of documents (i, j) is concordant if the rank ordering of the documents is the same in both lists and discordant if the rank ordering is different Kendall's Tau is defined as:

$$\tau = \frac{C}{C_n^2} \tag{3.7}$$

where C is the number of concordant pairs, and n is the total number of document pairs. The value of Kendall's Tau ranges from -1 to 1, where 1 indicates perfect agreement between the two rankings, -1 indicates perfect disagreement, and 0 indicates no correlation.
In this experiment, I categorized Kendall's τ values into four distinct intervals. Higher values of Kendall's τ suggest a stronger agreement or correlation between the predicted ranking and the ground truth ranking, indicating better performance and consistency of the ranking model. Conversely, lower values of Kendall's τ indicate less agreement or correlation, reflecting poorer performance and inconsistency in the rankings.
To provide a more detailed understanding, the τ values were divided as follows:

1. $\tau > 0.75$: This interval represents a very strong correlation, indicating that the predicted ranking is almost perfectly aligned with the ground truth.

2. $0.5 < \tau \leq 0.75$: Values in this range suggest a strong correlation, implying that the model's ranking is closely aligned with the actual relevance, albeit with some minor discrepancies.

3. $0.25 < \tau \leq 0.5$: This range indicates a moderate correlation, showing that while there is some agreement between the rankings, there are notable differences.
4. $0 < \tau \leq 0.25$: Values in this interval reflect a weak correlation, suggesting that the model's ranking does not align well with the ground truth, indicating significant inconsistencies.

In summary, by categorizing Kendall's tau values into these intervals, we can better evaluate and interpret the performance of the ranking model. High τ values, particularly those above 0.5, suggest that the model is performing well and producing rankings that are strongly correlated with the true relevance judgments. This approach provides a comprehensive framework for assessing the consistency and accuracy of the model's rankings, thereby ensuring appropriate performance evaluation.

3.5. System Requirements

3.5.1. Hardware

The process of data analysis and training of machine learning models requires significant computational power and storage resources. To execute the experiments in this study, I utilized a DESKTOP-A9EFOVM equipped with an Intel(R) Core(TM) i7-10875H CPU processor, a Google Colab V100 GPU, and 16GB of RAM.

3.5.2. Software

Python is selected as the programming language for this project due to several key advantages that align well with the demands of machine learning and data analysis. Its ease of use and robust support for object-oriented programming facilitate the development of complex algorithms and models. Furthermore, Python's compatibility with various popular libraries and frameworks significantly enhances its utility for scientific computing and machine learning tasks. The primary framework of the neural ranking GAM is implemented in PyTorch, taking advantage of its dynamic graph construction and extensive support for neural network operations. This project also leveraged the strengths of several notable libraries, which collectively contributed to its successful execution:

1. **BEIR (Benchmarking Information Retrieval):** This library provides a suite of benchmarks for evaluating information retrieval models across diverse datasets and tasks. BEIR offers standardized evaluation metrics and a comprehensive framework for comparing different retrieval methods, making it an invaluable tool for ensuring the robustness and generalizability of ranking models. In this project, BEIR facilitates rigorous evaluation and comparison of the neural ranking GAM against established benchmarks and provides the pre-trained black-box rankers for the training tasks.
2. **Pyesrini:** This library, serves as an information retrieval framework and provides efficient tools for data loading, preprocessing, and model evaluation. Pyserini supports sparse retrieval, dense retrieval (involves deep learning), and hybrid retrieval which integrates both approaches. In this project, we use Pyesrini to do the BM25 sparse retrieval.
3. **PyTorch:** This open-source machine learning library, developed by Facebook's AI Research lab (FAIR), is renowned for its dynamic computational graph and ease of use in building neural networks. PyTorch offers robust support for GPU acceleration, making it a preferred choice for deep learning applications. The fundamental framework of this project is developed using PyTorch, leveraging its flexibility and comprehensive ecosystem for implementing and training the neural ranking GAM.

4. **Scikit-Learn:** This free software machine learning library offers a comprehensive suite of tools for classification, regression, and clustering algorithms. Scikit-Learn's user-friendly interface and efficient implementations of numerous algorithms made it a cornerstone for model building and evaluation in this project.

5. **Matplotlib:** For data visualization, Matplotlib is a go-to library in the Python ecosystem. It enables the creation of high-quality plots and visualizations, which are vital for understanding data distributions, model performance, and results. Matplotlib's ability to integrate with GUI toolkits also allows for embedding plots into applications, enhancing their usability.

6. **NumPy:** As a foundational library for numerical computing in Python, NumPy provides powerful capabilities for working with large, multi-dimensional arrays and matrices. Its extensive collection of mathematical functions supports efficient array operations, essential for data preprocessing and numerical computations in machine learning workflows.

4 Results

Once the hyperparameters were appropriately configured, the models were trained and evaluated on their respective datasets and splits. This chapter presents a comprehensive analysis of the experimental results for the three tasks mentioned in Chapter 4.

4.1. Task 1

In this task, we evaluated the performance of our neural ranking Generalized Additive Models (GAMs) in explaining one single ranking output of black-box rankers on two datasets: SciFact and FiQA-2018. For each dataset, one query was randomly selected, and the top-20 documents retrieved by BM25 were reranked using the black-box rankers—ANCE for SciFact and ME for FiQA-2018. The effectiveness of our neural ranking GAMs was measured using Kendall's τ.

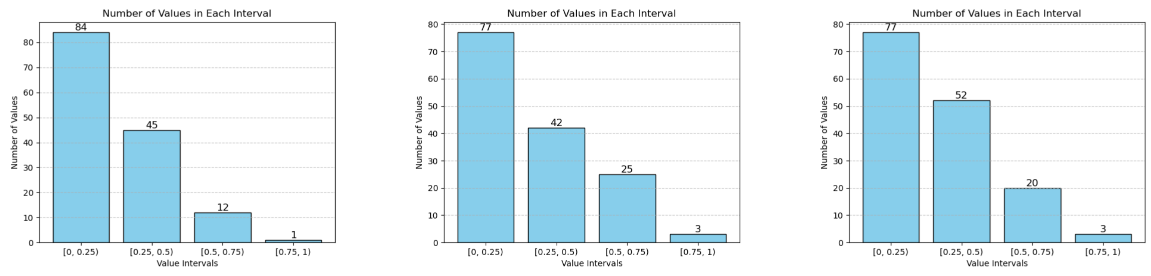
Both the SciFact and FiQA-2018 datasets presented similar outcomes in our experiment, allowing for a unified analysis. Throughout the training process, Kendall's τ was continuously monitored for both datasets. Remarkably, in both cases, **Kendall's τ value reached a perfect score of 1** after 60 training epochs and remained unchanged thereafter. This consistency indicates that the rankings produced by our neural ranking GAM models were identical to those generated by the respective black-box rankers (ANCE for SciFact and ME for FiQA-2018) for the given queries.

The achievement of Kendall's τ value of 1 across both datasets is highly significant. It demonstrates that our neural ranking GAM models have successfully captured and replicated the ranking logic of the black-box models, providing a perfect explanation for one single query.

4.2. Task 2

In Task 2, we exclusively utilize the SciFact dataset. For each query within the SciFact dataset, we select the top-10 documents for reranking based on the BM25 method. To streamline processing and enhance training efficiency, we adopt a batching strategy where eight queries are grouped into one batch, allowing for the simultaneous processing of multiple samples.

We started with constructing a global vocabulary comprising 4056 distinct candidate terms, encompassing all queries within this dataset, in accordance with the methodology delineated in Section 3.2. Subsequently, we proceed with the training and testing phases. Among the 1109 queries in the dataset, 887 are allocated for training purposes, with the remaining 222 designated for testing. Despite the implementation of batch training, the duration for one epoch remains at five hours, thereby limiting our training to three epochs. It is imperative to note that



(a) Epoch 1 (b) Epoch 2 (c) Epoch 3

Figure 4.1: The positive Kendall's τ value distribution in first three epochs

traditional Kendall's τ values are utilized here:

$$\tau = \frac{C - D}{C_n^2} \tag{4.1}$$

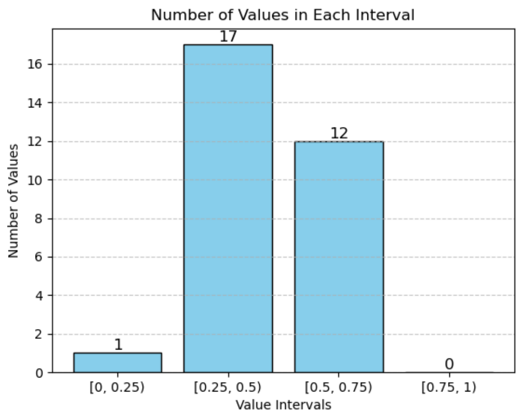
where C is the number of concordant pairs, D is the number of discordant pairs, and n is the total number of document pairs, so the negative values are possible. This serves as a main point for optimization in Task 3.
Across these three epochs, we observe an increase in the number of positive Kendall's τ values, with counts of 142, 147, and 152, respectively. Figure 4.1 illustrates the distribution of positive Kendall's τ values during the initial three epochs. Notably, there is a notable increase in the number of test results falling within the intervals $[0.25, 0.5)$ and $[0.5, 0.75)$ as the training epochs progress. The average Kendall's τ values for the three epochs were 0.256, 0.292, and 0.293, as shown in Table 4.1. This gradual increase in the average τ values indicates an improvement in the model's performance over the epochs. Although the improvement from the second to the third epoch appears marginal, the consistent increase from the first to the second epoch demonstrates that the model is learning and refining its ranking predictions to better align with the ground truth. Simultaneously, there is a corresponding increment in the count of positive values. This trend signifies that the predicted outcomes of the GAM model on the test set are exhibiting a heightened correlation with the ground truth.

Epoch	1	2	3
Average τ value	0.256	0.292	0.293

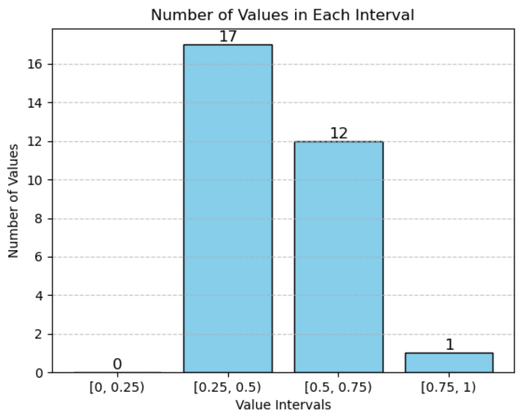
Table 4.1: Average τ value for each epoch in Task 2

Although Task 2 demonstrates that the model is being optimized through training, it is evident that the majority of test queries still fall into the lower τ value ranges, and the changes are not significantly pronounced. Furthermore, the training time for a single epoch precludes the possibility of conducting more epochs. Therefore, Task 3 includes optimizations for both the experimental setup and evaluation methods to address these issues.

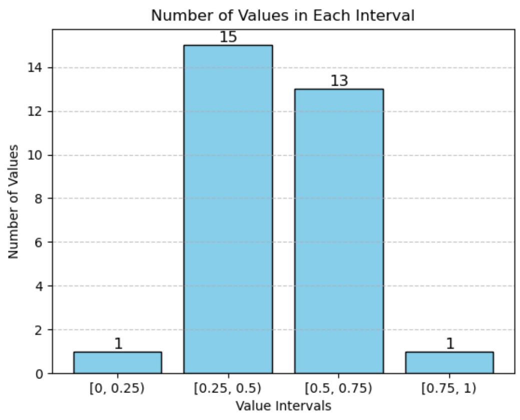
4.3. Task 3



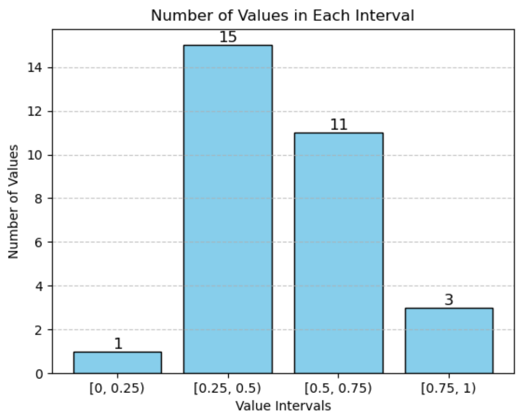
(a) Epoch 1



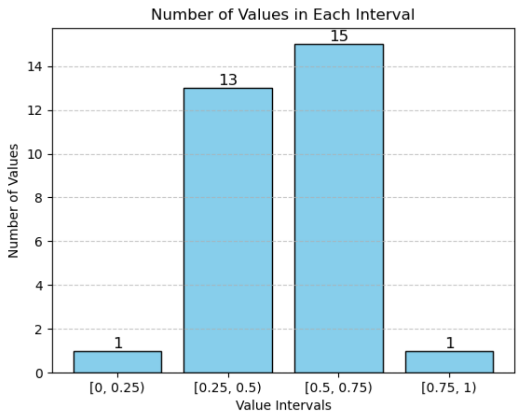
(b) Epoch 10



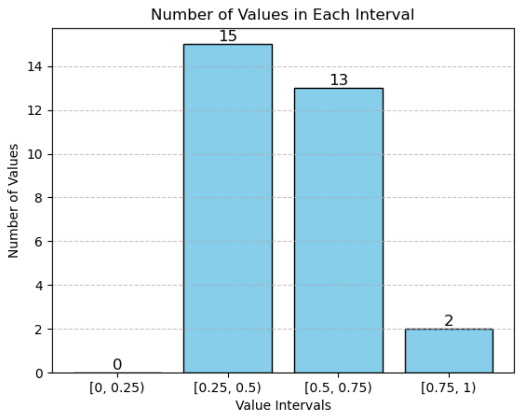
(c) Epoch 15



(d) Epoch 20



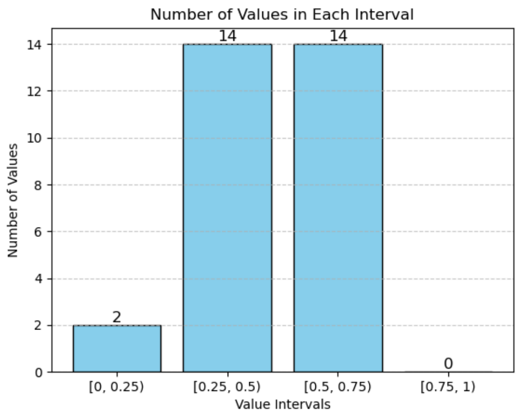
(e) Epoch 25



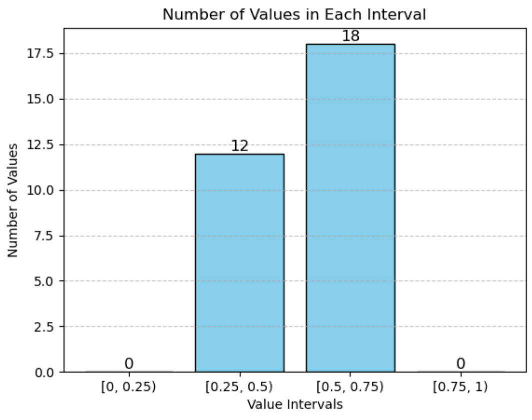
(f) Epoch 30

Figure 4.2: The Kendall's τ value distribution in first 30 epochs(SciFact+ANCE)

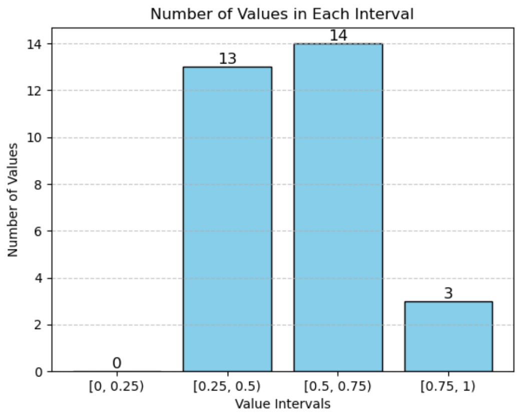
In Task 3, which represents the primary experimental work of this paper, we implemented several key optimizations to address the extended training times observed in Task 2. First, we randomly selected 150 queries from each dataset, with 120 queries for training and 30 for testing, to manage the dataset size more efficiently. To ensure document diversity, we retrieved the top-100 documents for each query based on the BM25 method and selected



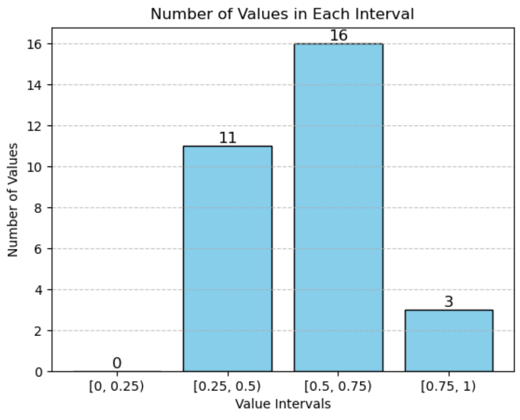
(a) Epoch 1



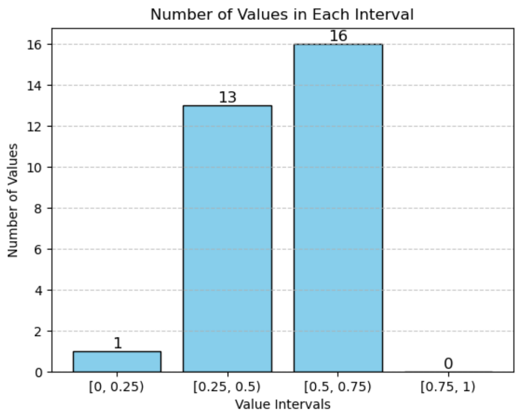
(b) Epoch 10



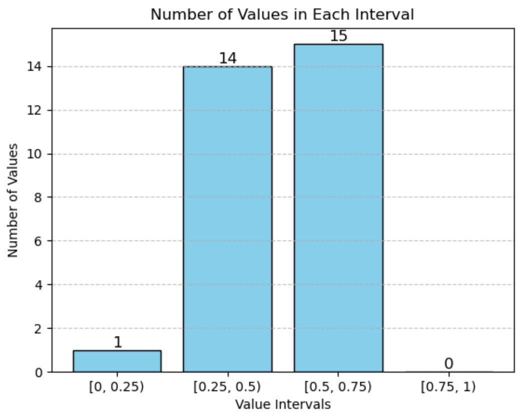
(c) Epoch 15



(d) Epoch 20



(e) Epoch 25



(f) Epoch 30

Figure 4.3: The Kendall's τ value distribution in first 30 epochs(SciFact+ME)

every tenth document for reranking. This approach ensures a broader spectrum of document relevance. Additionally, we increased the number of candidate terms per document from 5 to 10, providing the model with a richer set of features to enhance its ranking capabilities. We applied the methodology described in Section 3 to calculate Kendall's τ values and compared the $NDCG@10$ against the BM25 baseline. The test results for both datasets are shown as follows.

4.3.1. SciFact

ANCE Bi-Encoder Model Explanation:
For the ANCE bi-encoder model, we employed the ListNet loss function to optimize the model's performance. Throughout the training process, we continuously monitored and assessed the model's progress using Kendall's τ. Additionally, we implemented checkpoints at different stages of training to save models based on the loss incurred.
Kendall's τ value distribution in the first 30 epochs, as depicted in Figure 4.2. A $\tau > 0.5$ is considered as a strong correlation between the predicted ranking list and the original ranking list. Over the training epochs, we observed a decrease in the quantity of weak correlations, with more values entering the range above 0.5. Upon reaching epoch 30, half queries demonstrated a strong correlation among the 30 test sets evaluated. This observation shows our neural ranking GAMs well explain the original model.
ME Cross-Encoder Model Explanation:
For the ME cross-encoder model, we chose the MSE loss as the loss function. In comparison to the ANCE bi-encoder model, we observed larger fluctuations in the performance of this model. This variation could potentially be attributed to the change in loss function.
In Figure 4.3, we noted that the model's effectiveness peaked around the twentieth epoch, after which its performance gradually deteriorated. We attribute this phenomenon to various factors, including limited diversity, overfitting, and gradient instability:

1. **Limited Diversity:** Since we only use a subset of the dataset, small datasets often lack the diversity and variability necessary for robust model training. The model may struggle to capture the underlying patterns in the data, especially if the dataset is not representative of the broader population or lacks sufficient coverage of different scenarios or conditions.

2. **Overfitting:** With a limited amount of data, there's a higher risk of overfitting when using MSE loss. The model may learn to fit the training data too closely, capturing the features specific to the dataset rather than generalizable patterns. As a result, the model's performance on new, unseen data may lead to a worse performance.

3. **Gradient Instability:** For small datasets, the gradients computed using MSE loss may be less stable. Variations in the data can result in large fluctuations in the gradients, making it challenging for the optimization algorithm to converge to a satisfactory solution. This instability can hinder the model's ability to learn effectively.

Summary:

Ranker	ANCE + ListNet Loss	ME + MSE Loss
Average τ value	0.516	**0.551**

Table 4.2: Average τ value for two black-box rankers on the SciFact dataset

In Table 4.2, we show the best average Kendall's tau value in 30 epochs of the two black-box rankers on the SciFact dataset. We observe that using MSE loss resulted in a value of

0.551 while using ListNet loss resulted in a value of 0.516. Both results fall in the range of strong correlation, indicating good performance. Several factors may contribute to MSE loss performing better than ListNet loss on this dataset. First, MSE loss directly minimizes the squared error between predicted and true values, offering a clear optimization objective. This direct approach helps the model effectively learn ranking patterns, unlike ListNet loss, which involves complex sorting and softmax operations. Additionally, MSE loss is relatively insensitive to changes in model complexity and can adapt well to models of varying complexity. In ranking tasks, where the model may need to handle complex features and relationships, MSE loss provides a robust optimization framework that performs well across different model architectures. This robustness ensures that the model can achieve good performance regardless of its complexity. Furthermore, MSE loss is a straightforward and intuitive function, making it easier to interpret and understand the optimization process compared to ListNet loss rely on cross-entropy and softmax operations. These factors collectively make MSE loss a preferred choice in this task, resulting in superior performance and more reliable rankings.

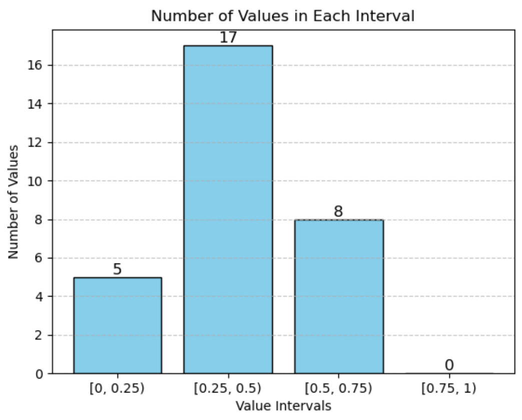
4.3.2. FiQA-2018

ANCE Bi-Encoder Model Explanation:
For the ANCE bi-encoder model, we employed the MSE loss function to optimize the model's performance. Kendall's τ value distribution in the first 30 epochs, as depicted in Figure 4.4. We observed that the model's performance did not exhibit significant improvement with increasing epochs. Even by epoch 30, only 10 files fall in the strong correlation range.
The discrepancy in model performance can be primarily attributed to the nature of relevance scores generated by the ANCE ranker. The use of dot products for scoring tends to produce larger values compared to cosine similarity-based scores. MSE loss function in this context poses challenges due to the inherent properties of dot product scores, which tend to magnify errors, resulting in exceptionally large loss values during training. This phenomenon makes it difficult for the model to converge to a reasonable solution within a feasible number of epochs. However, we utilized the ListNet Loss for the ANCE ranker on the SciFact dataet before and the model performance was much better. This difference in scale highlights the importance of employing appropriate normalization techniques, such as the softmax function within the ListNet Loss framework. The softmax function effectively handles the varying magnitudes of relevance scores, leading to more stable and reliable optimization in our previous work.
ME Cross-Encoder Model Explanation:
For the ME cross-encoder model, we chose the ListNet loss as the loss function. In comparison to the ANCE bi-encoder model, we observed the performance of this model is more stable and superior. As shown in Figure4.5, one observation is that over half of the queries consistently fall within the strong correlation range. However, compared to the SciFact dataset, there are relatively fewer τ values exceeding 0.75. This discrepancy might be attributed to the fact that the number of documents in the FiQA dataset is ten times larger than that in the SciFact dataset. Achieving a high level of consistency becomes significantly more challenging with such a vast corpus.
Summary:

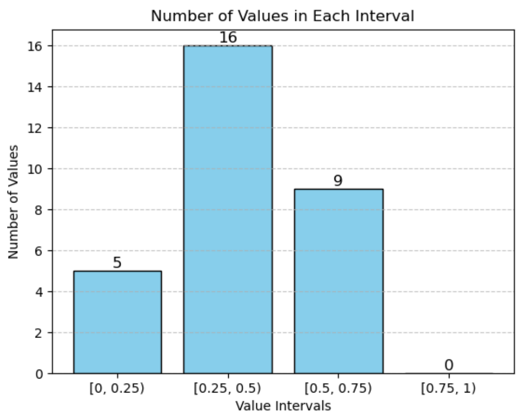
Ranker	ANCE + MSE Loss	ME + ListNet Loss
Average τ value	0.433	**0.531**

Table 4.3: Average τ value for two black-box rankers on the SciFact dataset

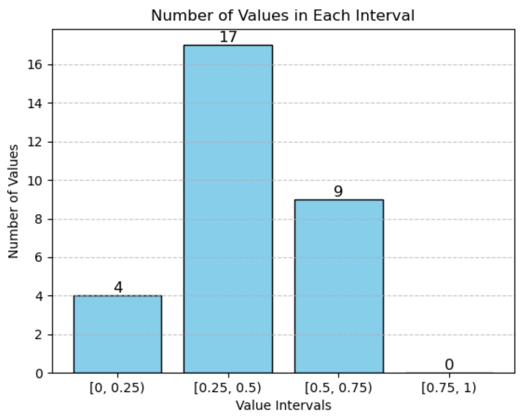
In Table 4.3, we show the best average Kendall's tau value in 30 epochs of the two black-box rankers on the FiQA-2018 dataset. We observe that using MSE loss combined with ANCE ranker resulted in a value of 0.433, this is also the only one among the four models with an average τ value below 0.5. The utilization of MSE Loss in the dot product-based ranking scores is not suitable. Additionally, refining the normalization and scaling techniques applied to the relevance scores could help mitigate the challenges associated with training convergence and loss optimization.



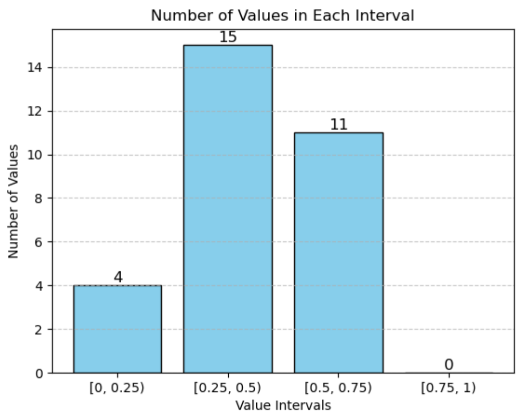
(a) Epoch 1



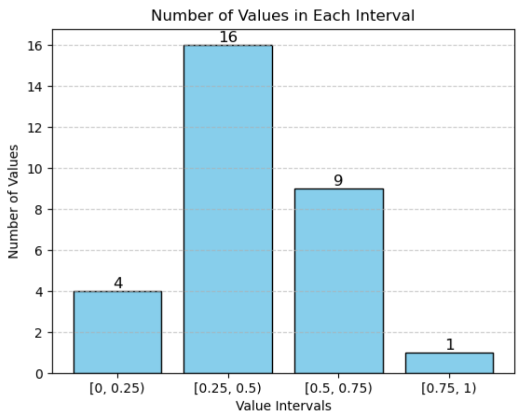
(b) Epoch 10



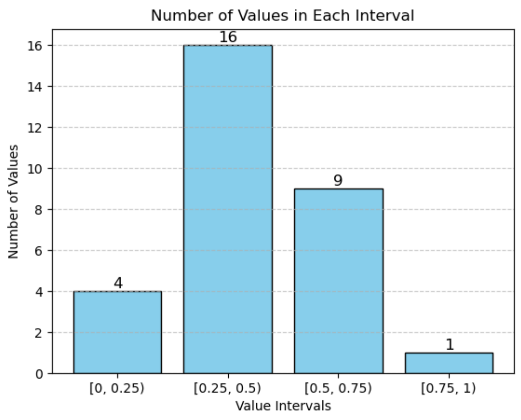
(c) Epoch 15



(d) Epoch 20

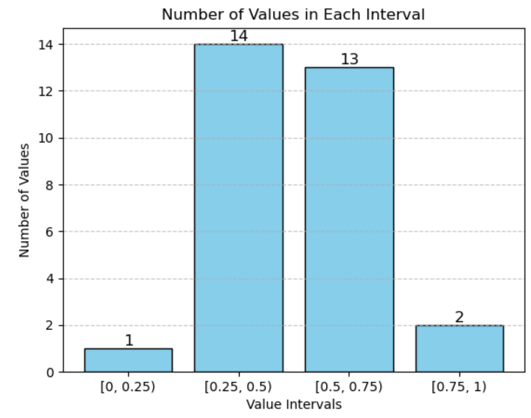


(e) Epoch 25

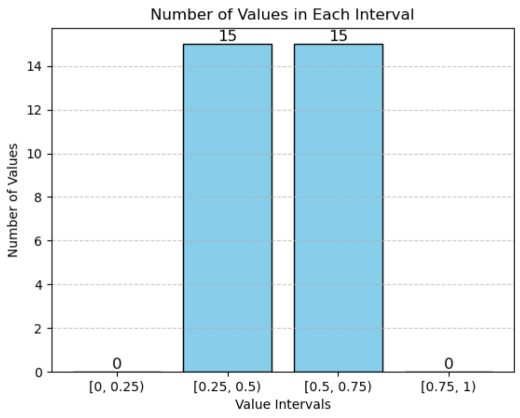


(f) Epoch 30

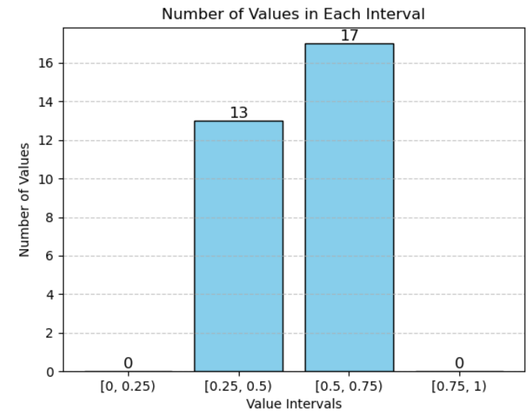
Figure 4.4: The positive Kendall's τ value distribution in first 30 epochs(FiQA-2018+ANCE)



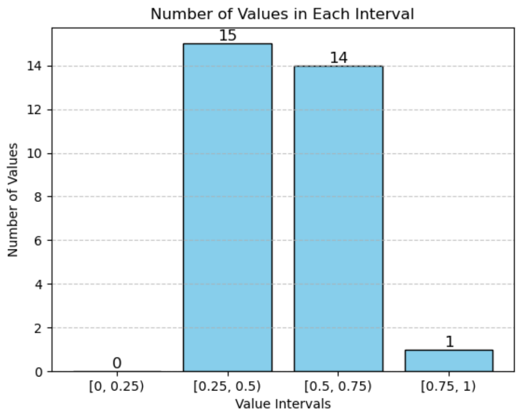
(a) Epoch 1



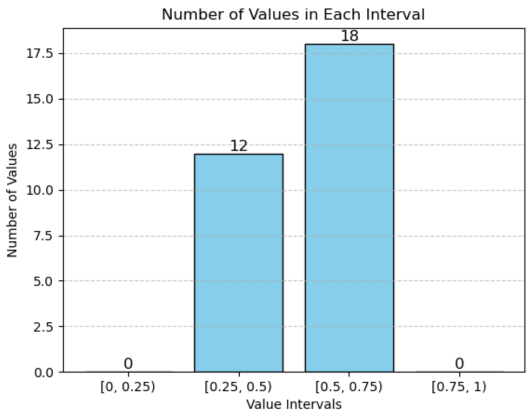
(b) Epoch 10



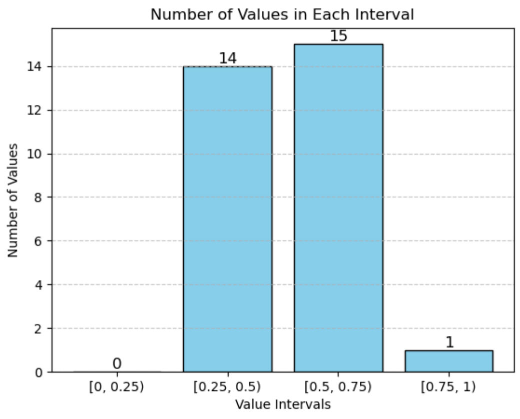
(c) Epoch 15



(d) Epoch 20



(e) Epoch 25



(f) Epoch 30

Figure 4.5: The positive Kendall's τ value distribution in first 30 epochs(FiQA-2018+ME)

4.3.3. NDCG Evaluation

Datasets	BM25	ANCE	ANCE+GAM	ME	ME+GAM
SciFact	48.7	48.7	25.6	47.4	18.3
FiQA-2018	15.5	15.9	7.15	17.16	10.69

Table 4.4: NDCG@10 of the BM25,Black-box Rankers,and GAMs

From Table 4.4, it can be observed that the NDCG@10 results of our model are worse than the BM25 baseline on both datasets. The observed disparity can be attributed to several factors related to the methodology employed in the evaluation. In this section, we will explain the reasons for this issue and propose feasible solutions to address it.
In our experiments, for each query, we selected 10 documents for reranking from the BM25 retrieval top-100 results by picking every 10th document to ensure diversity. However, the qrels in both datasets provide for each query with only one or two documents marked as most relevant. In this scenario where only one or two documents are deemed relevant, the NDCG@10 score is predominantly determined by whether these highly relevant documents appear within the reranking list. If these documents are absent from the documents we picked for the reranking task, the NDCG@10 score will be 0.
The NDCG results for each query provided by BM25 are generally either 1 or 0. This indicates that the results from BM25 either contain the most relevant document in the first position or do not contain it at all. This phenomenon occurs because we sample every tenth document, resulting in significant differences between documents. Essentially, if the first document in the reranking list is not the same as the qrels-provided ground truth, it is unlikely that any subsequent documents will be. This situation makes it challenging for our model to outperform BM25.
If we selected the top-10 documents in the retrieval results for the reranking task, our model's performance might improve. This is because the ground truth document is likely to be among these ten documents, allowing our model to rank the ground truth document higher, thereby achieving a higher NDCG@10 score than BM25. However, this approach would increase the difficulty of training our model, as the relevance differences among the top-10 documents are minimal. This rationale explains why we sample every tenth document.
To address this issue, we can select top-20 or top-30 documents for one query from the top-100 documents retrieved by BM25 for reranking and use these documents to train our model. This approach ensures both relevance and a sufficient quantity of documents to include the ground truth document. Subsequently, we can compute the NDCG results for the top-10 documents. This necessitates further optimization of our candidate term selection process. Currently, the vocabulary size for the ten documents within one query exceeds 3000 words, which complicates model simplification and extends training time.

5 Conclusions

This section addresses the three tasks presented in section 3.2 by analyzing the findings from the results. Subsequently, a more comprehensive examination of the limitations and future work is provided.

5.1. Conclusion

The experimental evaluation of the neural ranking interpretability models reveals insights into their efficacy across diverse tasks and datasets. Initially, the investigation focused on the performance of neural ranking Generalized Additive Models (GAMs) in explaining the ranking output of black-box rankers. Notably, the models achieved a perfect Kendall's τ score of 1 for one given query across both the SciFact and FiQA-2018 datasets, demonstrating their ability to replicate the ranking logic of the black-box models locally.

Subsequently, the model performance on a larger scale is evaluated, albeit with challenges such as limited dataset diversity and training time constraints. We implemented several optimization strategies to address these issues, including dataset size reduction and refinement of document selection methodologies, which is pivotal in enhancing model efficacy and addressing training time constraints. A comprehensive analysis of the GAMs on the SciFact and FiQA-2018 datasets was conducted, and three out of the four interpretability models achieved a strong correlation.

However, challenges such as fluctuating model performance and convergence difficulties were encountered, particularly with the ANCE bi-encoder model combined with the MSE Loss Function on the FiQA-2018 dataset. We analyzed this situation and proposed that when dealing with relatively large relevance scores, applying softmax to the ranking list can be beneficial. This allows for obtaining the probability distribution of the desired ranked documents, effectively controlling the relevance scores within a reasonable range.

5.2. Limitations

In Section 4.3.3, a critical observation concerning the disparity in NDCG@10 results compared to the BM25 baseline on both datasets. Due to our efforts to reduce the number of training documents while ensuring document diversity, we sampled every tenth document from the top 100 retrieved documents, resulting in lower NDCG values for our model. Moreover, focusing on only 150 queries from a dataset with thousands of queries, with only 30 used for testing, inevitably impacts the scalability performance of our model. Additionally, we conducted only one experiment for each of the four scenarios in Task 3 with time constraints, necessitating further assessment of the stability of the trained GAMs. Lastly, we observed that each vocabulary in Task 3 contains over 3000 keywords, with a significant portion of words in most documents likely to be included in this vocabulary, possibly ranging from 40% to 50%. Such a large number of keywords can result in an excessive number of subnetworks in our GAMs, leading to prolonged training times.

5.3. Future Work

Future research should focus on developing and evaluating novel sampling strategies tailored explicitly for the candidate term selection. These strategies aim to address the challenges associated with a large number of words in ranking and seek to enhance the scalability and efficiency of the neural ranking GAMs.

Another important direction for future work is to explore alternative list-wise loss functions. From the experiments, we can observe that the two loss functions we employed each have their drawbacks, which impact the effectiveness of model training.

Lastly, future studies need to assess the generalizability and robustness of the GAMs in ranking tasks. It is essential to evaluate its performance across diverse domains and datasets, like conducting experiments on different types of ranking scenarios, such as web search, recommendation systems, or social media ranking.

Bibliography

Boteva, V., Gholipour, D., Sokolov, A., Riezler, S., 2016. A full-text learning to rank dataset for medical information retrieval. URL: `http://www.cl.uni-heidelberg.de/~riezler/publications/papers/ECIR2016.pdf`.

Clark, K., Luong, M.T., Le, Q.V., Manning, C.D., 2020. Electra: Pre-training text encoders as discriminators rather than generators. `arXiv:2003.10555`.

Doshi-Velez, F., Kim, B., 2017. Towards a rigorous science of interpretable machine learning. `arXiv:1702.08608`.

Guo, J., Fan, Y., Pang, L., Yang, L., Ai, Q., Zamani, H., Wu, C., Croft, W.B., Cheng, X., 2019. A deep look into neural ranking models for information retrieval. `arXiv:1903.06902`.

Hastie, T., Tibshirani, R., 1986. Generalized additive models. Statistical Science 1, 297–310. URL: `http://www.jstor.org/stable/2245459`.

Letham, B., Rudin, C., McCormick, T.H., Madigan, D., 2015. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. The Annals of Applied Statistics 9. URL: `https://doi.org/10.1214%2F15-aoas848`, doi:`10.1214/15-aoas848`.

Liu, T.Y., et al., 2009. Learning to rank for information retrieval. Foundations and Trends® in Information Retrieval 3, 225–331.

Lundberg, S., Lee, S.I., 2017. A unified approach to interpreting model predictions. `arXiv:1705.07874`.

Lyu, L., Anand, A., 2023. Listwise explanations for ranking models using multiple explainers, in: European Conference on Information Retrieval, Springer. pp. 653–668.

Miller, T., 2018. Explanation in artificial intelligence: Insights from the social sciences. `arXiv:1706.07269`.

Molnar, C., 2022. Interpretable Machine Learning. 2 ed. URL: `https://christophm.github.io/interpretable-ml-book`.

Nawrocka, A., Kot, A., Nawrocki, M., 2018. Application of machine learning in recommendation systems, in: 2018 19th International Carpathian Control Conference (ICCC), pp. 328–331. doi:`10.1109/CarpathianCC.2018.8399650`.

Ribeiro, M.T., Singh, S., Guestrin, C., 2016. ”why should i trust you?”: Explaining the predictions of any classifier. `arXiv:1602.04938`.

Shrikumar, A., Greenside, P., Kundaje, A., 2017. Learning important features through propagating activation differences, in: International conference on machine learning, PMLR. pp. 3145–3153.

Singh, J., Anand, A., 2018a. Exs: Explainable search using local model agnostic interpretability. `arXiv:1809.03857`.

Singh, J., Anand, A., 2018b. Posthoc interpretability of learning to rank models using secondary training data. `arXiv:1806.11330`.

Singh, J., Anand, A., 2020. Model agnostic interpretability of rankers via intent modelling, in: Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, pp. 618–628.

Wadden, D., Lin, S., Lo, K., Wang, L.L., van Zuylen, M., Cohan, A., Hajishirzi, H., 2020. Fact or fiction: Verifying scientific claims, in: Webber, B., Cohn, T., He, Y., Liu, Y. (Eds.), Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online. pp. 7534–7550. URL: `https://aclanthology.org/2020.emnlp-main.609`, doi:`10.18653/v1/2020.emnlp-main.609`.

Xiong, L., Xiong, C., Li, Y., Tang, K.F., Liu, J., Bennett, P., Ahmed, J., Overwijk, A., 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. `arXiv:2007.00808`.

Zhuang, H., Wang, X., Bendersky, M., Grushetsky, A., Wu, Y., Mitrichev, P., Sterling, E., Bell, N., Ravina, W., Qian, H., 2020. Interpretable learning-to-rank with generalized additive models. `arXiv:2005.02553`.

Zytek, A., Arnaldo, I., Liu, D., Berti-Equille, L., Veeramachaneni, K., 2022. The need for interpretable features: Motivation and taxonomy. `arXiv:2202.11748`.