# FULTR: A Large-scale Prior-Posterior Fusion Learning to Rank Dataset and Its Application for Satisfaction-Oriented Ranking


Yuchen Li
Baidu Inc.
Beijing, China
yuchenli1230@gmail.com

Hao Zhang
Baidu Inc.
Beijing, China
haozhang.nku@gmail.com

Haojie Zhang
Baidu Inc.
Beijing, China
zhanghaojie03@baidu.com

Hengyi Cai
Baidu Inc.
Beijing, China
hengyi1995@gmail.com

Xinyu Ma
Baidu Inc.
Beijing, China
xinyuma2016@gmail.com

Shuaiqiang Wang
Baidu Inc.
Beijing, China
shqiang.wang@gmail.com

Haoyi Xiong
Baidu Inc.
Beijing, China
haoyi.xiong.fr@ieee.org

Zhaochun Ren
Leiden University
Leiden, The Netherlands
z.ren@liacs.leidenuniv.nl

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
leichen@cse.ust.hk

Dawei Yin*
Baidu Inc.
Beijing, China
yindawei@acm.org



## Abstract

The exponential growth of online content and increasingly diverse user needs have underscored the necessity for ranking models that go beyond traditional relevance assessments. Although several open-source benchmarks have significantly advanced academic research in Learning-to-Rank (LTR), these datasets predominantly focus on either text-based relevance or user behavior (click-through or dwell time) signals separately. This separation has inadvertently burdened academic progress by limiting the exploration of multi-faceted, satisfaction-oriented ranking models. In contrast, industry research has begun to delve into integrated approaches that fuse prior (relevance, authority, recency, and quality) with posterior (user interaction such as clicks and dwell time) signals, thereby better capturing true user satisfaction. In this paper, we introduce FULTR–a large-scale, prior-posterior FUsion LTR dataset. FULTR comprises over 224M queries and 683M documents from Baidu Search, combining both: (1) a rich prior-attribute set with detailed textual relevance, authority, recency, and quality features, and (2) a comprehensive posterior-attribute set enriched by user click data, dwell time, and positional information. By unifying these dual perspectives, FULTR establishes a robust, reproducible benchmark for satisfaction-oriented ranking, enabling researchers to develop models that better capture real-world search behaviors and user satisfaction. In addition, we propose a strong LTR baseline that merges a satisfaction ranker that leverages pre-trained language models to integrate diverse satisfaction signals, with a behavior ranker that captures user interactions using a dual-tower approach. Their outputs are combined via a fusion layer, yielding significant performance gains in multiple evaluation metrics, as confirmed by extensive experiments and ablation studies. We are confident that our contribution not only democratizes access to industrial-grade fusion data for the research community but also paves the way for more effective and holistic LTR model design. FULTR is available to the research community at https://github.com/zhanghao731/FULTR.



## CCS Concepts

• **Information systems** → **Learning to rank**.

## Keywords

Learning to Rank; Prior and Posterior Fusion Dataset; Web Search

**ACM Reference Format:**
Yuchen Li, Hao Zhang, Haojie Zhang, Hengyi Cai, Xinyu Ma, Shuaiqiang Wang, Haoyi Xiong, Zhaochun Ren, Maarten de Rijke, and Dawei Yin. 2025. FULTR: A Large-scale Prior-Posterior Fusion Learning to Rank Dataset and Its Application for Satisfaction-Oriented Ranking. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3711896.3737443



*Corresponding author.




*KDD '25, Toronto, ON, Canada*



ACM ISBN 979-8-4007-1454-2/2025/08
https://doi.org/10.1145/3711896.3737443


## 1 Introduction

Web search has evolved into an indispensable tool for information retrieval, and the challenge of ranking search results to maximize

user satisfaction has gained increasing attention [40]. Traditional learning-to-rank (LTR) methods predominantly focus on assessing the textual relevance between queries and webpages using models such as RankNet [3] and LambdaMART [37]. More recently, approaches leveraging pre-trained language models (PLMs) and attention mechanisms like BERT [9] and ERNIE [34] have enriched the field by capturing complex semantic nuances and cross-item interactions. In addition to developing advanced LTR models and algorithms [18–24], pioneering researchers have generously made a diverse array of LTR datasets publicly accessible to the research community. While open-source benchmarks such as ORCAS [7] and MS-MARCO [29] have provided a foundation for evaluating ranking performance, they typically focus on either human-annotated relevance judgments or user behavior signals–not both.

In standard LTR formulations, the goal is to produce an ordering of documents that maximizes relevance to a query. However, text-based relevance alone does not capture the full spectrum of user satisfaction. To address the need to fuse prior and posterior information for LTR, recent advances have begun incorporating user feedback such as click-through rates, dwell time, and other behavioral signals into ranking systems [4, 27, 41]. Although industry researchers have begun to explore this field, large-scale benchmarks and datasets that are publicly available to societies still treat human-annotated relevance and user behavior as separate streams [17, 43, 45, 46]. *Such separation restricts the ability of the academic research community to simulate the multifaceted nature of search satisfaction with prior (i.e., relevance) and posterior (i.e., user behavior) information.*

Specifically, existing efforts in open-source datasets and benchmarks could be categorized into three folders: (1) relevance-based retrieval, such as MS-MARCO [29], consisting of a large collection of user-generated queries and texts extracted from webpages or documents, (2) click-through prediction, such as ORCAS [7], usually containing records of user interactions with search results, and (3) LTR, such as Microsoft LETOR [30], including synthetic features of query-webpage pairs and human-annotated scores. To the best of our knowledge, no large-scale dataset currently offers a unified view of search-related data within LTR formulations by integrating webpage and query texts for relevance evaluation, user behavioral data, comprehensive satisfaction features, and well-annotated satisfaction scores. A closer examination of existing datasets reveals three key technical challenges that hinder the development of truly effective satisfaction-oriented ranking models:

- **Integrating Heterogeneous Signals**: Existing works typically consider textual relevance and user behavior independently. Although attention-based architectures and multivariate scoring functions have recently demonstrated the potential of integrating such diverse signals [11, 17], many models still lack a unified framework that seamlessly fuses prior (*i.e.*, human-annotated) and posterior (*i.e.*, user interaction-derived) information.
- **Data Fusion and Annotation Quality**: While curated datasets like LETOR deliver fine-grained relevance annotations, they are expensive and may not reflect the full variability of actual search behavior. Conversely, datasets based on click logs often contain noise and bias [6, 36]. The challenge lies in combining these two sources in a manner that leverages the high quality of expert labels alongside the richness of real-world user behavior, a problem that has been partially addressed by recent algorithms introducing pre-training techniques [17, 25, 26].
- **Robustness Across Real-World Scenarios**: Many benchmarking datasets fail to capture the diverse and dynamic nature of user interactions found in live search engines. Recent work has shown that addressing factors such as position bias and temporal dynamics is essential for building robust ranking models [1]. The lack of integrated datasets capable of accommodating these real-world variances hampers progress in this domain.

To overcome these challenges, we introduce FULTR, a large-scale prior-posterior FUsion LTR dataset designed explicitly for satisfaction-oriented web search. Our proposal addresses the integration of heterogeneous signals by constructing two complementary data components. The *prior component* captures detailed textual relevance, authority, recency, and quality features with high-quality human annotations. In contrast, the *posterior component* leverages massive user interaction logs encompassing click-through rates, dwell time, and additional behavior features. Furthermore, we propose a *fusion ranker* that contains a satisfaction ranker to model the prior signals, a behavior ranker to harness the posterior data, and a fusion layer to integrate both perspectives effectively. This design builds on recent advances in bivariate scoring functions that capture pairwise document relationships and attention-based models that successfully aggregate cross-item interactions. In comparison to existing open-source LTR datasets, FULTR provides a unique novelty that lies in its integrated prior-posterior fusion strategy. While many of these datasets provide valuable insights into either relevance or user behavior, they often do so in isolation. FULTR uniquely bridges this gap by providing a comprehensive benchmark that supports both perspectives—enabling researchers to develop and evaluate LTR models that more accurately capture user satisfaction in real-world search settings. In summary, our work makes several key technical contributions as follows:

- We present a large-scale dataset that fuses high-quality human-annotated relevance information with rich, real-world user interaction data, providing a dual perspective essential for benchmarking satisfaction-oriented ranking.
- We develop a strong LTR baseline with a three-component ranking framework that separately models and subsequently integrates prior and posterior signals, thereby addressing the challenges of heterogeneous information integration, data fusion quality, and real-world robustness.
- We enhance the current corpus of open-source resources by offering exhaustive documentation, comprehensive statistical analyses, and reproducible evaluation protocols that leverage state-of-the-art algorithms and datasets, fostering further research in this critical area.

By offering a comprehensive, open-source resource that combines both prior (*i.e.*, diverse satisfaction) and posterior (*i.e.*, user behavior) information, FULTR democratizes academic research in web search with industrial-grade state-of-the-art LTR. The proposed dataset empowers researchers and practitioners to benchmark novel ranking models that not only produce topically relevant

**Table 1: Comparison of FULTR to other publicly available relevance-oriented LTR datasets. This table shows the number of queries (#Q), documents (#D), language of search results (Lang.), and text attribute (Text) for each dataset. Relevance, Quality, Authority, and Recency are four types of prior-attribute features. Click, Dwell time (Dwell), and Position (Pos.) are three types of posterior-attribute features. Ann. represents the type of annotations. †datasets do not release user feedback and hide the original text of queries and documents, so researchers have to simulate click data.**

| Dataset | #Q | #D | Lang. | Text | Relevance | Quality | Authority | Recency | Click | Dwell | Pos. | Ann. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MS-MARCO [29] | 516K | 8.8M | EN | Token | ✓ | - | - | - | - | - | - | Binary |
| TREC CAR [10] | 2M | 30M | EN | Raw | ✓ | - | - | - | - | - | - | Binary |
| TriviaQA [15] | 95K | 650K | EN | Raw | ✓ | - | - | - | - | - | - | Binary |
| $T^2$Ranking [39] | 307K | 2.3M | CN | Raw | ✓ | - | - | - | - | - | - | Fine-grained |
| CWRCzech [35] | 2.7M | 8.4M | CZ | Raw | ✓ | - | - | - | ✓ | ✓ | ✓ | Fine-grained |
| Yahoo! LETOR [5] | 21K | 508K | EN | Token | ✓ | - | - | - | ✓† | - | - | Fine-grained |
| Microsoft LETOR [30] | 19K | 2.3M | EN | Token | ✓ | - | - | - | ✓† | - | - | Fine-grained |
| Istella LETOR [8] | 23K | 7.3M | EN | Token | ✓ | - | - | - | ✓† | - | - | Fine-grained |
| TripClick [32] | 1.6M | 2.3M | EN | Raw | ✓ | - | - | - | ✓ | - | ✓ | Fine-grained |
| ORCAS [7] | 10.4M | 1.4M | EN | Raw | ✓ | - | - | - | ✓ | - | - | Binary |
| Yandex-WSCD [33] | 21.1M | 70.3M | RUS | Token | ✓ | - | - | - | ✓ | - | ✓ | Click Labels |
| Sougou-QCL [44] | 0.5M | 9.0M | CN | Raw | ✓ | - | - | - | - | - | - | Click Labels |
| TianGong-PDR [38] | 70 | 11K | CN | Raw | ✓ | - | - | - | - | - | - | Fine-grained |
| Baidu-ULTR [46] | 383.4M | 1.3B | CN | Token | ✓ | - | - | - | ✓ | ✓ | ✓ | Fine-grained |
| Sogou-SRR [42] | 6K | 63K | CN | Raw | ✓ | - | - | - | - | - | ✓ | Fine-grained |
| mMarco-Chinese [2] | 516K | 8.8M | CN | Raw | ✓ | - | - | - | - | - | ✓ | Binary |
| DuReader [31] | 97K | 8.9M | CN | Raw | ✓ | - | - | - | - | - | - | Binary |
| Multi-CPR [28] | 303K | 303K | CN | Raw | ✓ | - | - | - | - | - | - | Binary |
| FULTR | 224M | 683M | CN | Raw | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | Fine-grained |

results but also closely align with true user satisfaction in dynamic search environments.

## 2 Related Work

Recent advancements in modern LTR methodologies heavily depend on the availability of large-scale, high-quality datasets, as state-of-the-art improvements are predominantly driven by the systematic development of such benchmarks. Existing available datasets in the LTR community primarily target relevance-oriented ranking tasks, with their construction methodologies bifurcated by the underlying approaches to relevance signal acquisition: (1) *human-annotated relevance judgments* (via human annotation), (2) *user behavior signal mining* (via leveraging user behavior data collected during the company's services).

Microsoft's MS-MARCO [29] is a widely adopted large-scale benchmark containing 516K user-generated question-based queries from Bing search logs and 8.8 million query-document pairs extracted from web documents, with human editors generating answers to establish binary relevance annotations. The TREC CAR dataset [10] utilizes topics, outlines, and paragraphs sourced from Wikipedia, which comprises 2 million queries and 30 million query-document pairs with binary relevance annotations. Moreover, the TriviaQA dataset [15] collects question-document pairs from several trivia and quiz-league websites and includes passages from Wikipedia and other web documents, which contains 95K queries and 650K query-document pairs with binary annotations. The Chinese dataset, $T^2$Ranking [39], contains 307K queries and over 2M fine-grained annotated query-document pairs. CWRCzech [35] proposes a Czech set for the relevance task, which comprises about 50k raw query-document pairs with fine-grained annotations. Moreover, Yahoo! LETOR [5], Microsoft LETOR [30], and Istella LETOR [8] are three commonly used datasets with synthetic features of query-webpage pairs and human-annotated relevance scores.

The compilation of large-scale click datasets predominantly relies on search engine logs, exemplified by established English resources supplemented with domain-specific collections, such as TripClick [32], which contains 1.3 million health search pairs. Recent contributions like Microsoft's ORCAS [7] expand this paradigm with 18.8 million query-document pairs. Non-English datasets exhibit distinct characteristics across linguistic and technical dimensions: Yandex-WSCD [33] captures 35 million Russian search sessions with query anonymization through proprietary encryption, while Chinese counterparts demonstrate tiered approaches: Sogou-QCL [44] leverages click-through signals across 537K queries and 9M web pages, TianGong-PDR [38] implements four-grade human evaluations on Sina News-derived passages, and Baidu-ULTR [46] achieves industrial-scale coverage with 1.2 billion query-document pairs anonymized via dictionary masking. However, Chinese datasets face scalability and quality challenges: Sogou-SRR [42] contains merely 6K queries despite rich metadata including screenshots and parse trees, while machine-translated resources like mMarco-Chinese [2] inherit MS-MARCO's limitations through automated conversion. Emerging solutions attempt mitigation: DuReader [31] from Baidu adopts MS-MARCO's question-based human answer paradigm, and Multi-CPR [28] focuses on multi-domain vertical search with binary relevance labels on result titles rather than full documents.

**Discussion.** Existing datasets mentioned above predominantly focus on relevance-oriented ranking tasks. However, with the exponential growth of web content, search users' demands have

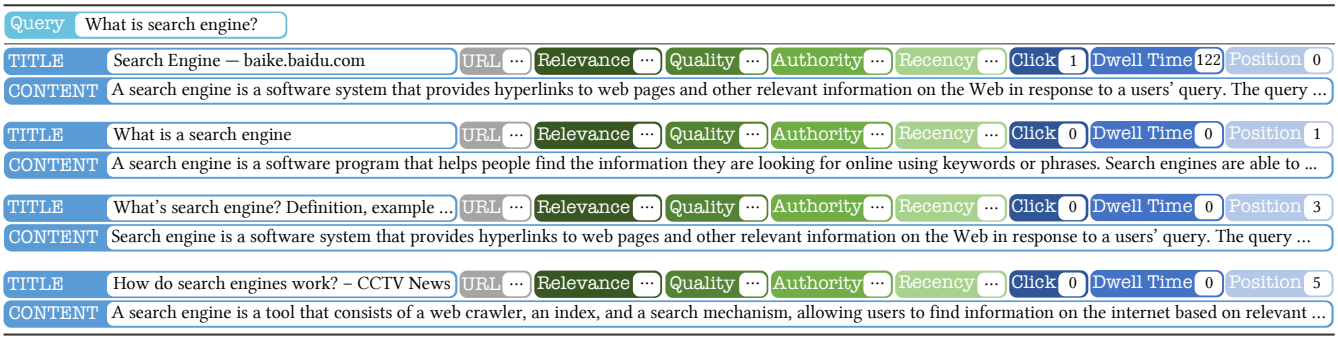



**Figure 1: Visualization of a query in FULTR, where each query-document pair is represented with textual, prior and posterior features. Queries and documents are translated into English for better understanding.**

diversified significantly; they now require not only high query-webpage relevance but also demand satisfactory results across multi-dimensional criteria, including authority, quality, and recency from search engines. Furthermore, the current practice of solely optimizing ranking outcomes through relevance-oriented prior attributes (from annotated datasets) or exclusively leveraging posterior attributes derived (from user behavior data) to enhance satisfaction proves inadequate. To address these limitations, we propose a large-scale dataset integrating both prior- and posterior-attribute, structured into two dedicated subsets. Concurrently, we design a fusion ranking architecture that synergistically combines prior knowledge and posterior signals to advance research in satisfaction-driven search ranking. Table 1 presents statistics summarizing the proposed dataset in comparison with the above LTR datasets.

## 3 FULTR

In this section, we formally introduce the proposed dataset FULTR, which consists of two components: a prior-attribute dataset and a posterior-attribute dataset. We first introduce the data collection process from the perspective of collection, encompassing sampling and anonymization. Then, we detail two subsets in FULTR: the prior-attribute dataset and the posterior-attribute dataset, and describe their feature construction and data distribution.

### 3.1 Data Collection

The raw data in FULTR is collected from Baidu Search[1], which has become the largest Chinese search engine globally by user population, archived documents, and queries served. Currently, Baidu Search orchestrates trillions of webpages archived and indexed for search, serves over three hundred million daily active users, and handles billions of queries per day.

**Scenarios.** We collect raw data across two distinct scenarios within the Baidu Search app on the PC and mobile sides. Given an input query, Baidu Search first needs to extract keywords or phrases from the query and recognize the user's intention, and then assess the similarity and relevance between the query and webpages, retrieving a number of relevant webpages from a database of trillions. Then, the search engine sorts the retrieved webpages through the ranking and re-ranking stage. Next, the search engine tops the most relevant webpages in the response to the query.

**Data Sampling.** Regarding the sampling process of the prior data, we conducted batch sampling and manual annotation over a two-year period. This sampling strategy was implemented because (1) the costs associated with human resources and annotation are substantial, and (2) it allows us to progressively cover a wider range of user search demands over time. For the sampling process of the posterior data, we collected samples over a one-month period, covering the previously mentioned two scenarios. In particular, frequent queries might be sampled multiple times, and each occurrence receives a distinct identifier, which ensures that the query distribution in FULTR reflects the same distribution as the online system, giving greater weight to frequent queries. For the candidate documents of each query in the posterior data, we only record the displayed ones to reduce the cost of storing logs. Documents not displayed are typically less informative, as users cannot provide feedback on them. Consequently, a logged search session usually contains only 10 documents per query, the number of results shown on a single page.

**Data Desensitization.** When constructing the dataset, our primary aim was to safeguard user privacy and prevent any disclosure of sensitive or personally identifiable information. To achieve this, we implemented a strict protocol: (1) Queries flagged as pornographic, obscene, or generated by bots were removed; (2) Only queries containing alphabetical characters were selected to avoid the accidental exposure of numerical data (*e.g.*, credit card numbers); (3) Sessions were not merged by user IDs, and each query was included only if it appeared in at least five unique requests within the specified time frame, ensuring robust anonymization. In addition, all data collected in FULTR has been thoroughly desensitized and encrypted, and comprehensive data protection measures were adopted throughout the experiment to minimize any risk of data leakage. It is important to emphasize that the dataset is intended exclusively for academic research and is not meant for any commercial purposes.

### 3.2 Prior-attribute Subset in FULTR

According to the variance in task-processing modalities and the structural divergence of feature-space representations, FULTR can be categorized into a prior-attribute subset and a posterior-attribute subset through systematic decoupling. In this section, we detail

[1]https://www.baidu.com

the description of the prior-attribute dataset in FULTR from the perspective of features, labels and data distribution.

**Prior Features.** The prior-attribute dataset contains 54,587 queries and 1,877,103 documents, where each query-document pair is represented with a series of textual and numerical features. Specifically, the columns of one sample in the prior-attribute dataset can be described as follows:

- *Query* refers to a textual feature, which undergoes processing operations such as correction of errors, completion of partial inputs, rewriting, and other query optimization steps.
- *URL* refers to the URL of the corresponding document.
- *Title* refers to the title of the document, which is classified terms or text segments from the document identified by Baidu Search.
- *Summary* refers to the content summary of the document, which is processed by Baidu search engine using a query-weighted summary extraction algorithm. In particular, content fields appear empty when webpages restrict search engine indexing.
- *Relevance* features consists of seven-dimensional discrete numerical features, such as *the longest ordered non-contiguous subsequence*, *entity matching*, *et al.*, to comprehensively capture full-document relevance information.
- *Quality* features contain four discrete numerical features, which include *the document quality score* (serving as a direct quality indicator), *predicted dwell time* (capturing granular quality signals from document content), *et al.*
- *Authority* features can be represented by query-agnostic features, which contain two textual features and five discrete numerical features, such as *site name*, *rate of the producer*, *et al.*
- *Recency* features comprises a textual feature and two discrete numerical features, such as *the difference between the document's creation date and the current search date*, *fresh of the query*, *et al.*
- *Click* features contains a 39-dimensional feature vector comprising normalized continuous numerical attributes, such as *the average dwelling time*, *average scroll speed*, *number of long-click*, *click-through rate*, *et al.*, to enhance user satisfaction modeling for ranking models.

More detailed descriptions of the above prior features can be found in Appendix A. Figure 1 illustrates an example of a ranking results record for a user query, where each displayed document is described with the aforementioned diverse types of features.

**Prior Labels.** Conventional web relevance search relies on ranking models trained and evaluated with query-document pairs annotated by professional annotators assigning relevance scores. Relevance labels are scaled from 0 to 4 to represent varying levels of relevance [30]. However, relying solely on relevance annotations is insufficient for satisfaction-driven ranking tasks. To address this limitation, FULTR employs professional annotators to assign satisfaction scores to chosen query-document pairs, thereby constructing a satisfaction dataset for training the ranking model. Therefore, following the settings of [17], grade satisfaction is labeled on a 0-4 scale to reflect different degrees of user satisfaction as (*i.e.*, {*0-bad, 1-fair, 2-good, 3-excellent, 4-perfect*}). Moreover, FULTR is split into the training set (including 45,727 queries and 1,493,673 documents) and the test set (including 8,860 queries and 383,430 documents).

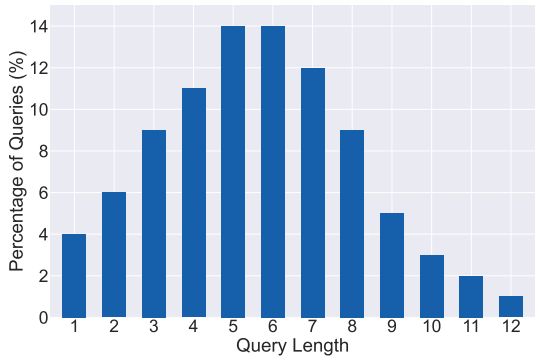



**Figure 2: The distribution of query length in words of the prior-attribute dataset.**

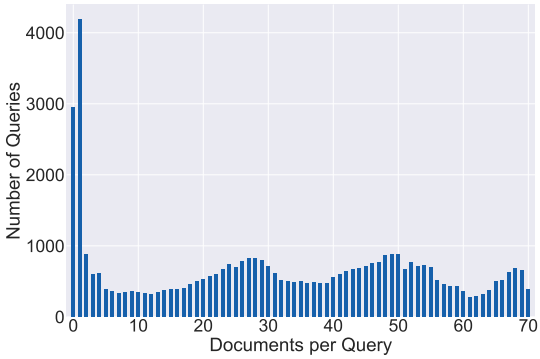
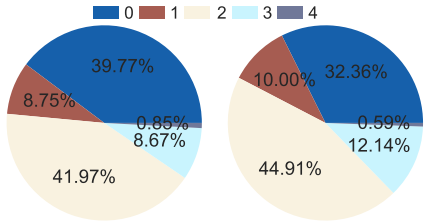



**Figure 3: The distribution of numbers of documents per query in the prior-attribute dataset.**




**(a) The training set.** **(b) The test set.**

**Figure 4: Pie chart of the annotation distribution of the prior-attribute dataset in FULTR.**

**Data Distribution and Analysis for Prior-Attribute Dataset.** In this section, we present some data analysis of the prior-attribute dataset in FULTR. Figure 2 illustrates the distribution of query length in words of FULTR. We tokenize the queries at the word level using the predefined vocabulary. According to the distribution, we could find that queries with lengths of 5 and 6 words are the most frequent, each accounting for approximately 14% of the total queries in the prior-attribute dataset. Moreover, we compute the number of documents per query in the prior-attribute dataset. As shown in Figure 3, the number of documents per query peaks at 1. Additionally, a non-negligible portion of queries lacks associated documents, a pattern consistent with real-world observations in

large-scale commercial search engines. Based on the annotation rules for satisfaction labels outlined in Section 3.2 and the division of the training and test sets, we separately calculate the proportion of diverse labels within both the training set and the test set. Figure 4 shows the distribution of the 5-level satisfaction annotation in the training set (a) and test set (b). Specifically, according to the statistical results, we could observe that samples labeled *2-good* and *0-bad* collectively accounted for over 70% in the training and test set, respectively. However, those with the highest satisfaction rating *4-perfect* constituted the smallest proportion. This is because our sampling process is conducted based on specific rules: it includes positive samples that rank high and negative samples that rank low. Moreover, negative samples and those with average scores (*i.e.*, *1* or *2*) comprise the majority of the overall dataset. Incorporating a large number of negative and average-score samples into the training data enhances the model's discriminative ability more effectively.

## 3.3 Posterior-attribute Subset in FULTR

Since complex feature constructions and costly professional annotations, necessary for a posterior-attribute dataset, are not required, labels for posterior data can be generated by checking the search log. Consequently, the posterior-attribute subset can be constructed on a significantly larger scale compared to the prior-attribute subset. In this section, we detail the posterior-attribute subset in FULTR, focusing on its features, labels, and data distribution.

**Posterior Features.** The prior-attribute dataset contains 224,427,699 queries and 681,416,004 documents, where a set of textual and numerical features characterizes each query-document pair. The posterior-attribute dataset comprises the following columns:

- *Query*, *URL*, *Title* and *Summary* are textual features, which are similar to the prior-attribute dataset, which represents the user query, the displayed URL, the displayed title of the document and the displayed summary of the document content.
- *Position* features refer to a discrete numerical feature, which is the displayed position of a document on the search result page.
- *User Behavior* features consist of 38-dimensional discrete numerical features and a continuous number feature. The click behavior of users and reading duration on candidate documents can reflect the degree of user satisfaction. Therefore, FULTR analyzes the search logs and designs a comprehensive set of user behavior features, including users' *query reformulations*, *the skip*, *the click*, *the first click*, *the last click*, *dwelling time*, *the display time on the screen*, *the displayed count on the screen*, *the slip-off count*, *et al.*

More detailed descriptions of the posterior features can be found in Appendix A.

**Posterior Labels.** User behaviors generally provide explicit indications of user satisfaction. For the posterior-attribute dataset, FULTR focuses on user behavior in terms of clicks and dwell time to design the posterior labels. Specifically, FULTR assigns a binary label for user click behaviour, where a document that the user clicks is 1 and 0 otherwise. Moreover, for the label of user dwell time, FULTR directly uses the duration of webpage browsing as the final score.

**Data Distribution and Analysis for Posterior-Attribute Dataset.** In this section, we present some data analysis of the posterior-attribute dataset in FULTR. Figure 5 shows the distribution of the number of documents per query in the prior-attribute dataset. Similarly to the prior-attribute dataset, the number of documents per query peaks at 1. Figure 6 shows the distribution of the number of clicks per search request. We could observe that over 70% of search requests exhibit singular click-through behavior, suggesting that a statistically significant majority of users achieve query resolution through a single interaction with search engine results pages. This phenomenon demonstrates optimal system performance in fulfilling information needs during initial engagement. Moreover, users' post-click behaviors (*i.e.*, dwell time) are equally significant. Furthermore, Figure 7 shows the kernel density estimation plot of


**Figure 5: The distribution of numbers of documents per query in the posterior-attribute dataset.**

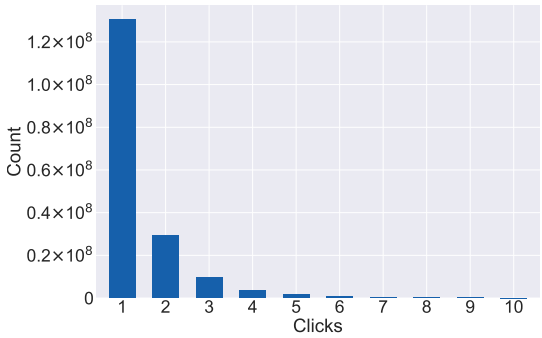



**Figure 6: The distribution of numbers of clicks per search request in the posterior-attribute dataset.**

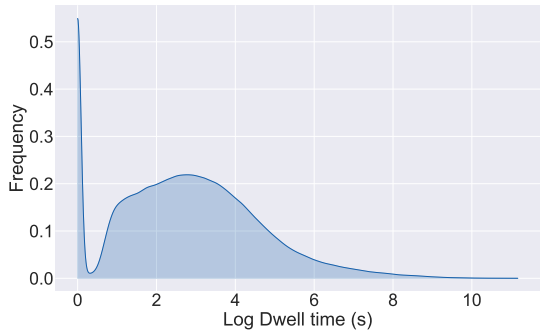



**Figure 7: Kernel density estimation plot of dwell time for clicked documents in the posterior-attribute dataset.**

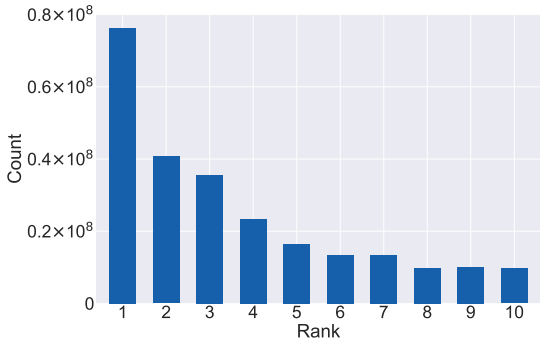



**Figure 8: The distribution of ranks for clicked documents in the posterior dataset.**

dwell time for clicked documents in the posterior-attribute dataset. We found that most clicked documents have relatively short dwell times, while a smaller proportion have much longer dwell times. Figure 8 shows the distribution of ranks for clicked documents in the posterior dataset. We observe that the probability of a click generally decreases with increasing rank, and more than half of the clicks occur in the top three results.

## 3.4 FULTR License

The dataset can be non-commercially used with a custom CC BY-NC 4.0 license[2]. In addition to the existing tasks in the dataset directory, users are permitted to create their own tasks under the license.

# 4 Methodology

In this section, we first formulate the satisfaction ranking task, and then propose a fusion ranker to demonstrate the potential contribution of FULTR for web satisfaction ranking tasks.

## 4.1 Satisfaction Ranking Task

The task of ranking aims to measure the relative order among a set of candidate documents $\mathcal{D} = \{d_i\}_{i=1}^{|\mathcal{D}|}$ under the constraint of a query $q \in \mathbb{Q}$, where $\mathcal{D} \subset \mathbb{D}$ is the set of $q$-related documents retrieved from indexed documents, and $d_i$ is the $i^{th}$ document retrieved for $q$. $\mathbb{Q}$ is the set of all given queries. For each document $d_i$, we assign a label $y_i \in \mathcal{Y}$ to indicate its satisfaction degree with respect to a query $q$, where $\mathcal{Y}$ is the set of all labels. The ranking model aims to learn a scoring function $f : \mathbb{Q} \times \mathbb{D} \rightarrow [0, 4]$ that maximizes the utility function: $\max_f \mathbb{E}_{\{q,\mathcal{D},\mathcal{Y}\}} \vartheta(\mathcal{Y}, F(q, \mathcal{D}))$, where $F(q, \mathcal{D}) = \{f(q, d_i)\}_{i=1}^{|\mathcal{D}|}$ represents the predicted ranking scores, and $\vartheta$ denotes evaluation metrics like NDCG [14] and PNR. Accordingly, we train the ranking model by minimizing the empirical loss over the labeled set:

$$\mathcal{L} = \frac{1}{|\mathcal{T}|} \sum_{q \in \mathbb{Q}} \ell(\mathcal{Y}, F(q, \mathcal{D})), \tag{1}$$

where $\ell$ represents the loss function comparing the predictions of the model for all retrieved documents in $\mathcal{D}$ for query $q$ against the ground truth labels.

[2]https://creativecommons.org/licenses/by-nc/4.0/

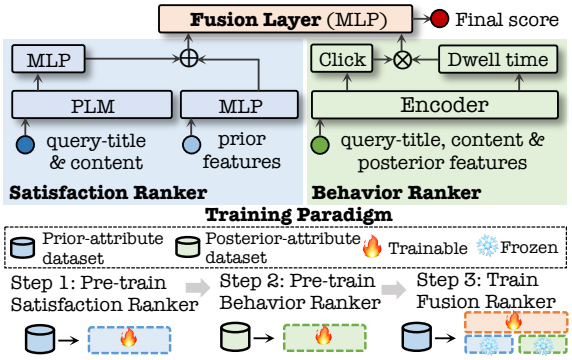



**Figure 9: The overall framework of Fusion Ranker and training paradigm using FULTR.**

## 4.2 Model Architecture

To empirically validate the efficacy of FULTR in user satisfaction-oriented ranking and to assess the distinct contributions of its two subsets, one tailored for the satisfaction-diverse ranking task and the other for the user behavior-based ranking task, we design three modular ranking architectures: *satisfaction ranker*, *behavior ranker*, and *fusion ranker*, to enable a systematic and multifaceted experimental evaluation.

**Satisfaction Ranker.** To model textual and diverse satisfaction features in FULTR simultaneously, we propose a hybrid model named *satisfaction ranker*, which consists of an ERNIE-based relevance extractor and an MLP-based satisfaction extractor. Specifically, to model the relevance among the textual input, given query-document pair $(q, d_i)$, *satisfaction ranker* first transforms $q$, the title and the content summary of $d_i$ into embeddings. Then, *satisfaction ranker* leverages an ERNIE-based encoder to learn the semantic relevance among the above textual inputs and generates a semantic representation. Next, given the learned representation, *satisfaction ranker* predicts the relevance score through an MLP. In the meanwhile, *satisfaction ranker* converts the numerical items (in *relevance*, *quality*, *authority*, and *recency* features) into embeddings and feeds textual items into a Transformer encoder to generate the textual representations. Next, *satisfaction ranker* concatenates generated embeddings and representations and feeds the combination into an MLP to compute a satisfaction score. Eventually, *satisfaction ranker* computes the summarization of the relevance score and the satisfaction score. In this way, the semantic relevance and diverse satisfaction are separately learned in *satisfaction ranker*. For the training phase, *satisfaction ranker* is optimized using the prior-attribute dataset in FULTR and minimizing a hybrid loss function combining pairwise and pointwise objective terms as

$$\begin{aligned} \ell(\mathcal{Y}, F(q, \mathcal{D})) = & \sum_{y_i < y_j} \max\left(0, f\left(q, d_i\right) - f\left(q, d_j\right) + \epsilon\right) \\ & + \lambda\left(\mu\left(f\left(q, d_i\right), y_i\right) + \mu\left(f\left(q, d_j\right), y_j\right)\right), \end{aligned} \tag{2}$$

where $\mu\left(f\left(q, d_i\right), y_i\right) = \max\left\{0, \left[f\left(q, d_i\right) - \left(\frac{y_i}{5} + 0.1\right)\right]^2 - \delta\right\}$ refers to the pointwise loss function, and $\epsilon$ refers to the manual margin enforced between positive and negative pairs. Furthermore, $\lambda$ and $\delta$ represent two hyper-parameters.

**Behavior Ranker.** As illustrated in Figure 9, to model textual and user behavior features in the posterior-attribute dataset of FULTR, we leverage a two-tower model named *behavior ranker*. Concretely, given a query-document pair $(q, d_i)$, *behavior ranker* first converts numerical features of posterior features into embeddings. Meanwhile, *behavior ranker* uses an ERNIE-based encoder to learn the semantic relevance among the above textual inputs and generates a semantic representation. Then, *behavior ranker* concatenates the embedding and representation and feeds the combination into a two-tower structure, which consists of a *click* tower and a *dwell time* tower to model click and reading duration, respectively. Next, *behavior ranker* feeds the outputs generated from two experts into two MLPs, and separately computes the predicted click score and dwell time score as $p_i^{click}$ and $t_i^{dwell}$. In this way, *behavior ranker* is trained using the posterior-attribute dataset and models the user behavior by minimizing the following loss function as

$$\mathcal{L}_{click} = -\sum_{i=1}^{|\mathcal{D}|} z_i \log p_i^{click} + (1 - z_i) \log \left(1 - p_i^{click}\right), \quad (3)$$

where $z_i$ is the label of the click task. Furthermore, *behavior ranker* models the dwell time by minimizing the duration loss function as

$$\mathcal{L}_{dwell} = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} (p_i^{click} t_i^{dwell} - t_i)^2, \quad (4)$$

where $t_i$ is the actual dwell time of $d_i$. Eventually, FULTR computes the sum of $\mathcal{L}_{dwell}$ and $\mathcal{L}_{click}$ as the final posterior loss $\mathcal{L}_{post}$: $\mathcal{L}_{post} = \alpha \mathcal{L}_{dwell} + \beta \mathcal{L}_{click}$, where $\alpha$ and $\beta$ are hyper-parameters to balance two loss functions.

**Fusion Ranker.** As shown in Figure 9, *fusion ranker* consists of a *satisfaction ranker*, a *behavior ranker*, and a *fusion layer*. Specifically, given the pre-trained *satisfaction ranker* and *behavior ranker*, *fusion ranker* combines their outputs and leverages a simple yet effective fusion layer (*i.e.*, an MLP) to fuse the output from *satisfaction ranker* and *behavior ranker* and estimate the final satisfaction score. In particular, during the training phase, *fusion ranker* is trained using the prior-attribute dataset with the frozen parameters in *satisfaction ranker* and *behavior ranker*. In this way, *fusion ranker* is optimized by minimizing a hybrid loss function combining pairwise and pointwise objective terms as Eq. (2).

# 5 Experiment

In this section, we conduct an empirical study of our proposed method and several baselines on FULTR.

## 5.1 Baselines

To fully demonstrate the potential contribution of FULTR for satisfaction ranking tasks, we select the following representative PLM-based ranking models as the baseline:

- **ERNIE-based Ranker** [34] is a vanilla ERNIE-based ranking model (*i.e.*, a 12-layer ERNIE with a pairwise loss), which is widely implemented and achieves advanced performance.
- **BERT-based Ranker** [12] is a vanilla BERT-based ranking model with a pairwise loss, which has been extensively employed in the LTR research community.
- **Fusion Ranker** refers to our proposed method, which separately models diverse satisfaction (prior) signals features and user behavior (posterior) features, and fuses the predicted results generated from two models.

Due to prior experience and the high costs associated with deploying suboptimal models, combined with our main contribution of being the first to propose a large-scale prior-posterior fusion LTR dataset, we only compare the aforementioned PLM-based rankers.

## 5.2 Evaluation Metrics

**Normalized Discounted Cumulative Gain (NDCG@K)** [14] is a standard listwise accuracy metric, which is widely used to evaluate the ranking model performance in the LTR community. Given a query and its corresponding documents, the ranking model usually estimates a score for each document and ranks them in descending order. The NDCG score for the query could be calculated as

$$\text{NDCG@}K = \frac{1}{I} \sum_{i=1}^{K} \frac{2^{y_i} - 1}{\log_2(1 + i)}, \quad (5)$$

where $I$ is a normalization factor for ideal Discounted Cumulative Gain (DCG) [13], and $y_i$ is the ranking score of the $i^{th}$ document.
**Positive-Negative Ratio (PNR)** is a commonly deployed pairwise metric to consider the partial order between labels in LTR research. For a query $q$ and its ranked documents $\mathcal{D}$, PNR can be calculated as the ratio of concordant pairs to discordant pairs as

$$PNR = \frac{\sum_{d_i, d_j \in \mathcal{D}} \mathbb{1}\left\{y_i > y_j\right\} \cdot \mathbb{1}\left\{f\left(q, d_i\right) > f\left(q, d_j\right)\right\}}{\sum_{d_m, d_n \in \mathcal{D}} \mathbb{1}\left\{y_m > y_n\right\} \cdot \mathbb{1}\left\{f\left(q, d_m\right) < f\left(q, d_n\right)\right\}}, \quad (6)$$

where $\mathbb{1}\{\cdot\}$ is an indicator that equals 1 if $(x > y)$, and 0 otherwise. PNR measures the alignment between the ground truth and the predicted ranking score.

## 5.3 Implementation Details

All the offline experiments are implemented on *PaddlePaddle Cloud* platform with 8 NVIDIA A100 GPUs. We adopt ERNIE-Lite as the backbone network and warm-initialize it for all ERNIE-based rankers. We set the number of layers, heads, and hidden dimensions of the PLMs to 12, 12 and 768. We configure the MLP layers in *fusion ranker* to 3 (*i.e.*, hidden layer of 512-256-128) and optimize the rankers with Adam [16].

## 5.4 Experimental Results

**Overall Comparison.** As illustrated in Table 2, we report the comparative results on PNR, NDCG@5 and NDCG@10 of selected PLM-based ranking models on FULTR. To conduct a fair comparison, ERNIE-based Ranker and BERT-based first convert the textual input and numerical features of both prior- and posterior-attribute features into embeddings. Then, two vanilla PLM-based ranking models concatenate these embeddings to construct the input for ranking tasks. According to the comparative results, we could find several phenomena as follows. First, *Fusion Ranker* consistently achieves the highest performance across all three metrics on our proposed dataset, which demonstrates its robustness on the real dataset. Concretely, *Fusion Ranker* achieves 3.623, 0.6567 and 0.7028 on PNR, NDCG@5 and NDCG@10, respectively. Compared to two vanilla PLM-based ranking models, it gains significant improvements on

**Table 2: Comparative results of PLM-based ranking models on FULTR. The best performance is highlighted in boldface.**

| Model | PNR | NDCG@5 | NDCG@10 |
|---|---|---|---|
| ERNIE-based Ranker | 3.362 ± 0.042 | 0.5847 ± 0.0026 | 0.6224 ± 0.0065 |
| BERT-based Ranker | 3.388 ± 0.039 | 0.5904 ± 0.0037 | 0.6371 ± 0.0023 |
| Fusion Ranker | **3.623** ± 0.015 | **0.6567** ± 0.0044 | **0.7028** ± 0.0028 |

**Table 3: Ablation study results of *Fusion Ranker* on FULTR.**

| Model | PNR | NDCG@5 |
|---|---|---|
| Fusion Ranker | 3.623 ± 0.045 | 0.7028 ± 0.0018 |
| Satisfaction Ranker (w/o posterior module) | 3.319 ± 0.024 | 0.6314 ± 0.0021 |
| Behavior Ranker (w/o prior module) | 3.260 ± 0.018 | 0.6219 ± 0.0039 |

FULTR. This phenomenon demonstrates that our proposed decoupled architecture achieves significant accuracy improvements on real-world data, compared to simultaneously modeling diverse satisfaction (*i.e.*, prior) signals and user behavior (*i.e.*, posterior) signals. Moreover, two vanilla PLM-based ranking models (*i.e.*, ERNIE-based Ranker and BERT-based Ranker) obtain strong performance on our proposed dataset, validating the effectiveness of FULTR for representative PLM-based ranking models and highlighting its potential contributions to satisfaction-oriented ranking tasks.

**Ablation Study of Model Structure.** To investigate the effectiveness of the key components in our proposed method, we carry out extensive ablation studies in this section. We consider the following variants: *Fusion Ranker* w/o posterior module (*i.e.*, Satisfaction Ranker) directly utilizes the hybrid model to learn the textual and numerical inputs, respectively. Moreover, *Fusion Ranker* w/o posterior module (*i.e.*, Behavior Ranker) concatenates the diverse satisfaction features with posterior features to construct the input for the two-tower model, and then leverages the click tower and the duration tower to model user click and duration behavior, respectively. Table 3 illustrates the ablation study results of removing two key components of *Fusion Ranker* on FULTR. According to the results, we could observe that two components benefit the proposed model, respectively. Particularly, removing the Behavior Ranker causes the sharpest drop on both metrics.

## 6 Conclusion

In this work, we propose a large-scale prior-posterior fusion LTR dataset FULTR, containing over 224M queries and 683M documents collected from Baidu Search. FULTR combines a rich prior-attribute set with diverse satisfaction features and a comprehensive posterior-attribute set enriched by user behavior information. By synthesizing these dual perspectives, FULTR establishes a robust and reproducible benchmark for satisfaction-oriented ranking, thereby enabling researchers to develop models that more accurately capture real-world search behaviors and user satisfaction. We propose a fusion ranker that combines a satisfaction ranker leveraging PLMs to integrate diverse satisfaction signals, a behavior ranker capturing user interactions via a dual-tower approach, and a fusion layer to integrate the output from two rankers. Extensive experiments and ablation studies show the effectiveness of our proposed model and the great potential to develop new algorithms with FULTR.

## Acknowledgments

We sincerely appreciate the reviewers for their insightful comments and valuable suggestions, and we extend our gratitude to all the engineers of the Baidu Search Science Team for their contributions to the construction of FULTR.

## References

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

[2] Luiz Bonifacio, Vitor Jeronymo, Hugo Queiroz Abonizio, Israel Campiotti, Marzieh Fadaee, Roberto Lotufo, and Rodrigo Nogueira. 2021. mmarco: A multilingual version of the ms marco passage ranking dataset. *arXiv preprint arXiv:2108.13897* (2021).

[3] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*. 89–96.

[4] Maarten Buyl, Paul Missault, and Pierre-Antoine Sondag. 2023. Rankformer: Listwise learning-to-rank using listwide labels. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3762–3773.

[5] Olivier Chapelle and Yi Chang. 2011. Yahoo! learning to rank challenge overview. In *Proceedings of the learning to rank challenge*. PMLR, 1–24.

[6] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2023. Bias and debias in recommender system: A survey and future directions. *ACM Transactions on Information Systems* 41, 3 (2023), 1–39.

[7] Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. ORCAS: 18 Million Clicked Query-Document Pairs for Analyzing Search. *CoRR* abs/2006.05324 (2020). arXiv:2006.05324

[8] Domenico Dato, Claudio Lucchese, Franco Maria Nardini, Salvatore Orlando, Raffaele Perego, Nicola Tonellotto, and Rossano Venturini. 2016. Fast ranking with additive ensembles of oblivious and non-oblivious regression trees. *ACM Transactions on Information Systems (TOIS)* 35, 2 (2016), 1–31.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 4171–4186.

[10] Laura Dietz, Manisha Verma, Filip Radlinski, and Nick Craswell. 2017. TREC Complex Answer Retrieval Overview.. In *TREC*.

[11] Liming Gao, Dongliang Liao, Gongfu Li, Jin Xu, and Hankz Hankui Zhuo. 2022. Semantic IR fused Heterogeneous Graph Model in Tag-based Video Search. In *Companion Proceedings of the Web Conference 2022*. 94–98.

[12] Shuguang Han, Xuanhui Wang, Mike Bendersky, and Marc Najork. 2020. Learning-to-Rank with BERT in TF-Ranking. *arXiv preprint arXiv:2004.08476* (2020).

[13] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[14] Kalervo Järvelin and Jaana Kekäläinen. 2017. IR evaluation methods for retrieving highly relevant documents. In *ACM SIGIR Forum*, Vol. 51. 243–250.

[15] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551* (2017).

[16] Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

[17] Canjia Li, Xiaoyang Wang, Dongdong Li, Yiding Liu, Yu Lu, Shuaiqiang Wang, Zhicong Cheng, Simiu Gu, and Dawei Yin. 2023. Pretrained Language Model based Web Search Ranking: From Relevance to Satisfaction. *arXiv preprint arXiv:2306.01599* (2023).

[18] Yuchen Li, Haoyi Xiong, Linghe Kong, Jiang Bian, Shuaiqiang Wang, Guihai Chen, and Dawei Yin. 2024. GS2P: a generative pre-trained learning to rank model with over-parameterization for web-scale search. *Machine Learning* (2024), 1–19.

[19] Yuchen Li, Haoyi Xiong, Linghe Kong, Zeyi Sun, Hongyang Chen, Shuaiqiang Wang, and Dawei Yin. 2023. MPGraf: a Modular and Pre-trained Graphformer for Learning to Rank at Web-scale. In *IEEE International Conference on Data Mining, ICDM 2023, Shanghai, China, December 1-4, 2023*. IEEE, 339–348. https://doi.org/10.1109/ICDM58522.2023.00043

[20] Yuchen Li, Haoyi Xiong, Linghe Kong, Qingzhong Wang, Shuaiqiang Wang, Guihai Chen, and Dawei Yin. 2023. $S^2$phere: Semi-Supervised Pre-training for Web Search over Heterogeneous Learning to Rank Data. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*. ACM, 4437–4448. doi:10.1145/3580305.3599935

[21] Yuchen Li, Haoyi Xiong, Linghe Kong, Shuaiqiang Wang, Zeyi Sun, Hongyang Chen, Guihai Chen, and Dawei Yin. 2023. Ltrgcn: Large-scale graph convolutional networks-based learning to rank for web search. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 635–651.

[22] Yuchen Li, Haoyi Xiong, Linghe Kong, Rui Zhang, Fanqin Xu, Guihai Chen, and Minglu Li. 2023. MHRR: MOOCs Recommender Service With Meta Hierarchical Reinforced Ranking. *IEEE Trans. Serv. Comput.* 16, 6 (2023), 4467–4480. https://doi.org/10.1109/TSC.2023.3325302

[23] Yuchen Li, Haoyi Xiong, Qingzhong Wang, Linghe Kong, Hao Liu, Haifang Li, Jiang Bian, Shuaiqiang Wang, Guihai Chen, Dejing Dou, and Dawei Yin. 2023. COLTR: Semi-Supervised Learning to Rank With Co-Training and Over-Parameterization for Web Search. *IEEE Trans. Knowl. Data Eng.* 35, 12 (2023), 12542–12555. https://doi.org/10.1109/TKDE.2023.3270750

[24] Yuchen Li, Haoyi Xiong, Yongqi Zhang, Jiang Bian, Tianhao Peng, Xuhong Li, Shuaiqiang Wang, Linghe Kong, and Dawei Yin. 2025. RankElectra: Semi-supervised Pre-training of Learning-to-Rank Electra for Web-scale Search. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining, V.1, KDD 2025, Toronto, ON, Canada, August 3-7, 2025*. 2415–2425. https://doi.org/10.1145/3690624.3709395

[25] Yuchen Li, Hao Zhang, Yongqi Zhang, Hengyi Cai, Mingxin Cai, Shuaiqiang Wang, Haoyi Xiong, Linghe Kong, Dawei Yin, and Lei Chen. 2025. RankExpert: A Mixture of Textual-and-Behavioral Experts for Multi-Objective Learning-to-Rank in Web Search. In *SIGKDD*.

[26] Yuchen Li, Hao Zhang, Yongqi Zhang, Xinyu Ma, Wenwen Ye, Naifei Song, Shuaiqiang Wang, Haoyi Xiong, Dawei Yin, and Lei Chen. 2025. M2oERank: Multi-Objective Mixture-of-Experts Enhanced Ranking for Satisfaction-Oriented Web Search. In *2025 IEEE 41st International Conference on Data Engineering (ICDE)*. IEEE Computer Society, 4441–4454. https://doi.ieeecomputersociety.org/10.1109/ICDE65448.2025.00333

[27] Congcong Liu, Yuejiang Li, Jian Zhu, Fei Teng, Xiwei Zhao, Changping Peng, Zhangang Lin, and Jingping Shao. 2022. Position awareness modeling with knowledge distillation for CTR prediction. In *Proceedings of the 16th ACM Conference on Recommender Systems*. 562–566.

[28] Dingkun Long, Qiong Gao, Kuan Zou, Guangwei Xu, Pengjun Xie, Ruijie Guo, Jian Xu, Guanjun Jiang, Luxi Xing, and Ping Yang. 2022. Multi-cpr: A multi domain chinese dataset for passage retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 3046–3056.

[29] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human-generated machine reading comprehension dataset. (2016).

[30] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 datasets. *arXiv preprint arXiv:1306.2597* (2013).

[31] Yifu Qiu, Hongyu Li, Yingqi Qu, Ying Chen, Qiaoqiao She, Jing Liu, Hua Wu, and Haifeng Wang. 2022. DuReader_retrieval: A large-scale chinese benchmark for passage retrieval from web search engine. *arXiv preprint arXiv:2203.10232* (2022).

[32] Navid Rekabsaz, Oleg Lesota, Markus Schedl, Jon Brassey, and Carsten Eickhoff. 2021. TripClick: The Log Files of a Large Health Web Search Engine. In *The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2507–2513.

[33] Pavel Serdyukov, Georges Dupret, and Nick Craswell. 2014. Log-based personalization: The 4th web search click data (WSCD) workshop. In *Proceedings of the 7th ACM international conference on Web search and data mining*. 685–686.

[34] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 8968–8975.

[35] Josef Vonásek, Milan Straka, Rostislav Krč, Lenka Lasonová, Ekaterina Egorova, Jana Straková, and Jakub Náplava. 2024. Cwrczech: 100m query-document czech click dataset and its application to web relevance ranking. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1221–1231.

[36] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to rank with selection bias in personal search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 115–124.

[37] Qiang Wu, Christopher JC Burges, Krysta M Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *Information Retrieval* 13 (2010), 254–270.

[38] Zhijing Wu, Jiaxin Mao, Yiqun Liu, Jingtao Zhan, Yukun Zheng, Min Zhang, and Shaoping Ma. 2020. Leveraging passage-level cumulative gain for document ranking. In *Proceedings of the web conference 2020*. 2421–2431.

[39] Xiaohui Xie, Qian Dong, Bingning Wang, Feiyang Lv, Ting Yao, Weinan Gan, Zhijing Wu, Xiangsheng Li, Haitao Li, Yiqun Liu, et al. 2023. T2ranking: A large-scale chinese benchmark for passage ranking. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2681–2690.

[40] Haoyi Xiong, Jiang Bian, Yuchen Li, Xuhong Li, Mengnan Du, Shuaiqiang Wang, Dawei Yin, and Sumi Helal. 2024. When Search Engine Services Meet Large Language Models: Visions and Challenges. *IEEE Trans. Serv. Comput.* 17, 6 (2024), 4558–4577. https://doi.org/10.1109/TSC.2024.3451185

[41] Tao Yang, Chen Luo, Hanqing Lu, Parth Gupta, Bing Yin, and Qingyao Ai. 2022. Can clicks be both labels and features? Unbiased Behavior Feature Collection and Uncertainty-aware Learning to Rank. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*. 6–17.

[42] Junqi Zhang, Yiqun Liu, Shaoping Ma, and Qi Tian. 2018. Relevance estimation with multiple information sources on search engine result pages. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 627–636.

[43] Yunan Zhang, Le Yan, Zhen Qin, Honglei Zhuang, Jiaming Shen, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2023. Towards disentangling relevance and bias in unbiased learning to rank. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 5618–5627.

[44] Yukun Zheng, Zhen Fan, Yiqun Liu, Cheng Luo, Min Zhang, and Shaoping Ma. 2018. Sogou-qcl: A new dataset with click relevance label. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 1117–1120.

[45] Honglei Zhuang, Zhen Qin, Xuanhui Wang, Michael Bendersky, Xinyu Qian, Po Hu, and Dan Chary Chen. 2021. Cross-positional attention for debiasing clicks. In *Proceedings of the Web Conference 2021*. 788–797.

[46] Lixin Zou, Haitao Mao, Xiaokai Chu, Jiliang Tang, Wenwen Ye, Shuaiqiang Wang, and Dawei Yin. 2022. A large scale search dataset for unbiased learning to rank. *Advances in Neural Information Processing Systems* 35 (2022), 1127–1139.

# A Detailed Feature Description

In this section, we provide a detailed description of the features of the prior-attribute dataset and the posterior-attribute dataset mentioned in Section 3.2 and 3.3. Table 4 presents the specific feature representation of samples in the prior-attribute dataset. Specifically, each query-document pair in the prior-attribute dataset is represented with 68-dimensional features and a label, which contains rich text features and diverse satisfaction features (*i.e.*, authority, recency, quality, and relevance). As the prior-attribute dataset is used to train the whole fusion ranker, it has click features the same as the click features in the posterior-attribute dataset. Moreover, Table 5 details the specific feature representation of samples in the posterior-attribute dataset. Due to restrictions on commercial disclosure and to prevent exposing user privacy, we have processed all original data in a privacy-preserving manner.

**Table 4: The description of features of the prior-attribute dataset in FULTR.**

| Attribute | No. | Feature Symbol | Feature Type | Explanation |
|---|---|---|---|---|
| Label | 1 | Label | Discrete Number | [0-4] |
| Text Features | 2 | qid | Uint64 | A search request is uniquely identified. |
| | 3 | query | Text | The search query. |
| | 4 | url | Text | The URL of the document. |
| | 5 | title | Text | The title of the document. |
| | 6 | click_query | Text | The set of search queries that have clicked on the URL. |
| | 7 | summary | Text | The summary of the document content. |
| | 8 | requirement | Text | The intent of the query. |
| | 9 | domain | Text | The industry classification of the query. |
| Authority Features | 10 | site_name_affiliated_organization | Text | The name of the website's affiliated organization. |
| | 11 | site_name_website | Text | The name of the website. |
| | 12 | producer_rate | Discrete Number | The level of the producer. |
| | 13 | domain_pr | Discrete Number | The level of the domain. |
| | 14 | doc_auth | Discrete Number | The author of the document. |
| | 15 | author_fans | Discrete Number | The number of fans of the author. |
| | 16 | is_official_website_site | Discrete Number | Whether it is an official website. |
| Recency Features | 17 | query_search_time | Text | The time of the search request. |
| | 18 | fresh_day | Discrete Number | The time gap between the search time and the document generation time. |
| | 19 | query_fresh | Discrete Number | The freshness level of the search query. |
| Quality Features | 20 | doc_len | Discrete Number | The length of the document. |
| | 21 | qual_score | Discrete Number | The quality level of the document. |
| | 22 | qual_label | Discrete Number | The quality label of the document. |
| | 23 | dt_pred | Discrete Number | The predicted dwell time. |
| Relevance Features | 24 | bhs_global_sum_term_qimp | Discrete Number | The sum of qimp values for all matches in the document. |
| | 25 | bhs_content_max_sequence_in_order_ratio | Discrete Number | The longest ordered (but non-consecutive) subsequence. |
| | 26 | bhs_content_most_important_concept_hit_num | Discrete Number | The number of matches in the content field for the largest fragment obtained by concatenating basic-level segments that have a tightness greater than 0.75. |
| | 27 | bhs_global_most_important_concept_hit_num | Discrete Number | The sum of qimp values for all matches in the document. |
| | 28 | bhs_global_query_perfect_hit | Discrete Number | Whether all the query terms appear consecutively and in order in the document (in either the title or content field). |
| | 29 | entity_match | Discrete Number | For the core entity in the query, a value of 1 is assigned if it is completely matched in the document, otherwise 0. |
| | 30 | bhs_content_window_6_max_sum_qimp | Discrete Number | In the content field, the qimp value of query terms hit within a window whose size is 32 times the query length. |
| Click Features | 31 | query_search | Discrete Number | The search frequency of the query. |
| | 32 | Hourly click through rate | Continuous Number | Percentage of clicks per impression for the given hour (clicks/impressions). |
| | 33 | Hourly normalized impression count | Continuous Number | Total impressions (times shown) in the hour, scaled to a standardized range (e.g., 0-1). |
| | 34 | Hourly click-review rate | Continuous Number | Ratio of clicks that underwent human or automated review (e.g., fraud detection) within the hour. |
| | 35 | Hourly normalized review count | Continuous Number | Number of reviewed clicks (after quality checks) in the hour. |
| | 36 | Hourly in-platform click-review rate | Continuous Number | Percentage of clicks that were both triggered and reviewed within the platform's ecosystem (e.g., app/internal pages). |
| | 37 | Hourly normalized average click-to-view duration | Continuous Number | Average time (normalized) between a user clicking and starting to actively view content (e.g., page load delay). |
| | 38 | Hourly normalized average click duration | Continuous Number | Average time (normalized) a user stays engaged after a click (e.g., reading time). |
| | 39 | Hourly normalized in-platform impression | Continuous Number | Number of impressions served within the platform (e.g., app/browser) during the hour. |
| | 40 | Daily click-through | Continuous Number | Daily aggregated CTR (clicks/impressions over 24 hours). |
| | 41 | Daily normalized impression count | Continuous Number | Total daily impressions, normalized for comparability across days. |
| | 42 | Daily click-review rate | Continuous Number | Daily ratio of clicks reviewed for quality/validity. |
| | 43 | Daily normalized review count | Continuous Number | Daily count of reviewed clicks. |
| | 44 | Daily in-platform click-review rate | Continuous Number | Daily proportion of in-platform clicks that were reviewed. |
| | 45 | Daily normalized average click-to-view duration | Continuous Number | Daily average delay between click and active viewing. |
| | 46 | Daily normalized average click duration | Continuous Number | Daily average time spent post-click. |
| | 47 | Daily normalized in-platform impression count | Continuous Number | Daily in-platform impressions. |
| | 48 | Weekly in-platform click-to-impression rate | Continuous Number | Ratio of in-platform clicks to impressions over the week. |
| | 49 | Weekly normalized average click-to-view duration | Continuous Number | Weekly average delay (click-to-view). |
| | 50 | Weekly normalized average click duration | Continuous Number | Weekly average post-click engagement time. |

**Table 4: The description of features of the prior-attribute dataset in FULTR (Continued).**

| Attribute | No. | Feature Symbol | Feature Type | Explanation |
|---|---|---|---|---|
| Click Features | 51 | Weekly normalized average view duration | Continuous Number | Weekly average time users actively viewed content (post-click). |
| | 52 | Weekly normalized impression count | Continuous Number | Weekly total impressions. |
| | 53 | Weekly normalized average logarithmic click duration | Continuous Number | Logarithmic transformation of weekly average click durations (compresses outliers). |
| | 54 | Weekly normalized average logarithmic view duration | Continuous Number | Logarithmic transformation of weekly average view durations. |
| | 55 | Weekly normalized average squared view duration | Continuous Number | Squared weekly average view durations. |
| | 56 | Weekly normalized average squared click duration | Continuous Number | Squared weekly average click durations (emphasizes longer durations). |
| | 57 | Weekly normalized standard deviation of click duration | Continuous Number | Weekly variability (spread) in click durations. |
| | 58 | Weekly normalized standard deviation of view duration | Continuous Number | Weekly variability in view durations. |
| | 59 | Monthly in-platform click-to-impression rate | Continuous Number | Monthly ratio of in-platform clicks to impressions. |
| | 60 | Monthly normalized average click-to-view duration | Continuous Number | Monthly average delay (click-to-view). |
| | 61 | Monthly normalized average click duration | Continuous Number | Monthly average post-click engagement time. |
| | 62 | Monthly normalized average view duration | Continuous Number | Monthly average view duration. |
| | 63 | Monthly normalized impression count | Continuous Number | Monthly impressions. |
| | 64 | Monthly normalized average logarithmic click duration | Continuous Number | Logarithmic monthly average click duration. |
| | 65 | Monthly normalized average logarithmic view duration | Continuous Number | Logarithmic monthly average view duration. |
| | 66 | Monthly normalized average squared click duration | Continuous Number | Squared monthly average click duration. |
| | 67 | Monthly normalized average squared view duration | Continuous Number | Squared monthly average view duration. |
| | 68 | Monthly normalized standard deviation of click duration | Continuous Number | Monthly variability in click durations. |
| | 69 | Monthly normalized standard deviation of view duration | Continuous Number | Monthly variability in view durations. |

**Table 5: The description of features of the posterior-attribute dataset in FULTR.**

| Attribute | No. | Feature Symbol | Feature Type | Explanation |
|---|---|---|---|---|
| Label | 1 | Label | Discrete Number | 0/1 |
| | 2 | qid | Uint64 | A search request is uniquely identified. |
| | 3 | ranking position | Discrete Number | The document's displaying order on the screen. |
| | 4 | query | Text | The search query. |
| | 5 | url | Text | The URL of the document. |
| | 6 | title | Text | The title of the document. |
| | 7 | summary | Text | The summary of the document content. |
| Click Features | 8-45 | The same as Feature No.32~No.69 in Table 4 | | |
| | 46 | dwell time | Continuous Number | The length of time a user spends looking at a document after they've clicked a link on a SERP page, but before clicking back to the SERP results. |