Can LLM Annotations Replace User Clicks for Learning to Rank?

Lulu Yu
State Key Laboratory of AI Safety, ICT, Chinese Academy of Sciences
University of Chinese Academy of Sciences
Beijing, China
yululu23s@ict.ac.cn

Keping Bi*
State Key Laboratory of AI Safety, ICT, Chinese Academy of Sciences
University of Chinese Academy of Sciences
Beijing, China
bikeping@ict.ac.cn

Jiafeng Guo
State Key Laboratory of AI Safety, ICT, Chinese Academy of Sciences
University of Chinese Academy of Sciences
Beijing, China
guojiafeng@ict.ac.cn

Shihao Liu
Shuaiqiang Wang
Dawei Yin
Baidu Inc.
Beijing, China
liushihao@baidu.com
wangshuaiqiang@baidu.com
yindawei@acm.org

Xueqi Cheng
State Key Laboratory of AI Safety, ICT, Chinese Academy of Sciences
University of Chinese Academy of Sciences
Beijing, China
cxq@ict.ac.cn

Abstract

Large-scale supervised data is essential for training modern ranking models, but obtaining high-quality human annotations is costly. Click data has been widely used as a low-cost alternative, and with recent advances in large language models (LLMs), LLM-based relevance annotation has emerged as another promising annotation. This paper investigates whether LLM annotations can replace click data for learning to rank (LTR) by conducting a comprehensive comparison across multiple dimensions. Experiments on both a public dataset, TianGong-ST, and an industrial dataset, Baidu-Click, show that click-supervised models perform better on high-frequency queries, while LLM annotation-supervised models are more effective on medium- and low-frequency queries. Further analysis shows that click-supervised models are better at capturing document-level signals such as authority or quality, while LLM annotation-supervised models are more effective at modeling semantic matching between queries and documents and at distinguishing relevant from non-relevant documents. Motivated by these observations, we explore two training strategies—data scheduling and frequency-aware multi-objective learning—that integrate both supervision signals. Both approaches enhance ranking performance across queries at all frequency levels, with the latter being more effective. Our code is available at https://github.com/Trustworthy-Information-Access/LLMAnn_Click.

*Corresponding author.

Conference acronym 'XX, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

CCS Concepts

• **Information systems → Retrieval models and ranking**.

Keywords

Unbiased Learning to Rank, Large Language Models, Relevance Annotation, Learning to Rank

ACM Reference Format:
Lulu Yu, Keping Bi, Jiafeng Guo, Shihao Liu, Shuaiqiang Wang, Dawei Yin, and Xueqi Cheng. 2018. Can LLM Annotations Replace User Clicks for Learning to Rank?. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 12 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Learning to rank (LTR) plays a crucial role in real-world scenarios, especially in search engines. The goal of LTR is to train a ranking model that can sort documents based on their relevance to a given query. Early LTR models are typically tree-based models such as LambdaMART [9], which rely on hand-crafted features extracted from query-document pairs, including traditional LTR features like BM25 [35]. With the rapid progress of deep learning—particularly the emergence of Transformer architectures [44]—modern ranking models are often implemented as cross-encoders [6, 47], which jointly encode the query and document. However, these models involve a large number of parameters, making them data-hungry during training. Since obtaining large-scale human annotations is expensive, alternative sources of low-cost supervision have been explored. Click data, readily available in real-world search logs, has emerged as a practical source of relevance annotations.

Search logs contain abundant user interaction signals, among which click signals are the most straightforward and widely used. Clicks serve as implicit indicators of relevance between queries and documents, and they can be collected at scale with low cost. However, click signals are inherently biased [18] due to various factors such as position bias [5, 24, 46], selection bias [2, 42, 43], and

Comparison Dimension	Click Data	LLM Annotations
Source of Annotated Documents	Fixed search engine results	Unrestricted
Annotation Granularity	Binary (click / no click)	Flexible (instruction-based)
Annotation Mechanism	Implicit, based on query intent	Instruction-driven (given annotation guidelines / ranking / selection)
Volume of Annotated Data	Large	Large
Annotation Cost	Low	Moderate (between click and manual annotation), lower with open-source LLMs, e.g., Qwen-series.
Basis for Relevance Judgment	**Semantic matching between queries and documents + document-side signals (e.g., quality, authority)**	**Semantic matching between queries and documents**
Annotation Challenges	Sparse clicks, false negatives, various biases	False positives

Figure 1: Comparison of annotation characteristics between click data and LLM annotations

trust bias [29, 45, 54]. This has led to extensive research in Unbiased learning to rank (ULTR), which aims to mitigate these biases and obtain unbiased ranking models. Due to the lack of publicly available real-world click datasets, most ULTR methods are evaluated on simulated click data, where clicks are generated according to assumed user browsing behavior [34, 41]. In contrast, real-world clicks result from more complex and nuanced user browsing behavior [19, 48, 56]. Therefore, developing effective methods to debias real-world click data and obtain unbiased ranking models remains a significant challenge.

With the rapid development of large language models (LLMs), using LLMs for relevance annotation has become increasingly promising [16, 33, 39, 52]. Like click data, LLM annotations also offer a low-cost alternative for relevance annotation compared to costly human annotations. Considering that LLM annotations tend to have lower noise than click data, it is natural to ask whether LLM annotations could potentially replace click data for LTR. To explore this question, we first conduct a comparative analysis of their annotation characteristics, as illustrated in Fig. 1. The most significant differences lie in two aspects: 1) LLMs provide greater flexibility in generating annotations; and 2) the criteria for relevance judgment differ. Click data can capture not only the semantic matching between queries and documents, but also document-level signals such as authority and quality. In contrast, LLM annotations are highly effective at capturing semantic relevance between query-document pairs, but are less sensitive to document-level signals. We design three annotation strategies using LLMs: generating multi-level annotations based on given annotation guidelines, directly ranking results, and having LLMs directly select relevant results. Experimental results show that, under the given annotation guidelines, using listwise input of query-document pairs for annotation leads to the best model performance.

Considering that search engines perform better on high-frequency queries [56], returning more relevant results, it is crucial to further distinguish relevance using document-level signals. Therefore, we conduct a comparative analysis of ranking models trained on click data and LLM annotations across queries of different frequencies, and further explore their capabilities by introducing LTR features from different levels (query-document level and document-level) [4, 10, 35, 50], shedding light on the complementarity between LTR features and ranking models trained with either click data or LLM annotations. The experiments demonstrate that models trained on click data perform better on high-frequency queries, while models trained on LLM annotations have an advantage on medium- and low-frequency queries. The experiments involving the introduction of different levels of LTR features further confirm that models trained on click data could capture document-level signals but are less effective in capturing semantic matching between queries and documents compared to models trained on LLM annotations. This suggests that query-document semantic features help address the limitations of click-trained models, while document-level signals can enhance the performance of models trained with LLM annotations.

Moreover, given the high proportion of relevant results in the training set, it is unclear whether the ranking models truly possess the ability to distinguish between relevant and irrelevant results. To address this, we introduce true negative samples to investigate the relevance discrimination capabilities of ranking models trained on these two types of annotations. The experiments indicate that models trained on LLM annotations have a stronger ability to distinguish between relevant and irrelevant results compared to those trained on click data. Additionally, introducing true negative samples into the training set improves the ability of models trained on click data to distinguish between relevant and irrelevant results, and further enhances their capability to differentiate the degree of relevance among relevant results.

Based on the comparative analysis above, click data and LLM annotations each have their advantages for queries of different frequencies. To leverage the strengths of both, we propose two methods to combine click data and LLM annotations: 1) Data Scheduling: since LLM annotations are of higher quality and have a stronger ability to capture semantic matching between queries and documents—which is fundamental for relevance judgment—we use the ranking model trained on LLM annotations for warm start. To prevent forgetting during the training process, we further perform interleaved training using click data from high-frequency queries and LLM annotations from medium- and low-frequency queries; 2) Frequency-Aware Multi-Objective Learning: we use query frequency as the gating signal to adaptively adjust the weights of both click data and LLM annotations during simultaneous training. Both of these combination methods achieve the goal of leveraging the respective advantages of each, but the latter performs better.

In summary, our contributions are:

- We perform a detailed empirical comparison of models trained with click data and LLM annotations from multiple dimensions.
- We find that models trained on click data can capture semantic matching and document-level signals, with performance advantages in high-frequency queries. Models trained on LLM annotations exhibit a stronger ability to capture semantic matching, with performance advantages in medium- and low-frequency queries, and are better at distinguishing between relevant and irrelevant content compared to those trained on clicks.
- We propose two hybrid training strategies that integrate both clicks and LLM annotations to leverage the complementary strengths of the two supervision signals.

2 Related Work

2.1 Learning to Rank with Click Data

As ranking models grow in parameter size, Learning to Rank (LTR) tasks require increasingly large amounts of training data, yet large-scale human annotations are costly. Click data, as an implicit feedback signal, offers a low-cost alternative but suffers from inherent noise and biases, motivating research on Unbiased Learning to Rank (ULTR) [4, 12, 26, 53]. Existing methods fall into two main groups. One group is click modeling [13, 14]. It formalizes user browsing behavior and estimates relevance by maximizing the likelihood of the observed clicks. However, such statistical approaches require frequent repetition of query–document pairs to ensure reliable estimation, making them ineffective for long-tail queries. The other group derives from counterfactual learning. It treats bias as a counterfactual factor and mitigates it via inverse propensity weighting (IPW) [24, 46], where the key challenge is tto estimate the propensity accurately For instance, Dual Learning Algorithm (DLA) [4] jointly trains unbiased ranking models and unbiased propensity models. Early studies primarily address position bias by modeling propensity solely as a function of rank position. In practice, however, biases are more complex: the click behavior for a result can be influenced by surrounding results, and in real-world systems, strong underlying ranking models often cause relevance–position coupling, propensity overestimation and severe false-negative issues. Zhang et al. [53] proposed observation dropout and gradient reversal for relevance–position decoupling, Luo et al. [26] introduced unconfounded propensity estimation to address propensity overestimation, and Yu et al. [48] proposed a new click hypothesis and the DualIPW to mitigate false negatives.

2.2 LLM-Based Annotation for Information Retrieval

With the continuous development of large language models (LLMs) [28, 38], their increasingly strong semantic understanding has sparked growing interest in using LLMs for automated annotation [17, 22, 25]. In the information retrieval (IR) domain, there are two major definitions of relevance that require annotation: relevance, which measures the inherent semantic relevance between queries and documents; and utility, a higher-level annotation that considers the contribution of documents to answering user queries beyond relevance. Thomas et al. [39] investigated how to design annotation prompts that enable LLMs to produce high-quality relevance judgments, demonstrating that LLM annotations can achieve accuracy comparable to that of crowd workers. Meanwhile, Zhang et al. [52] explored LLMs' ability to distinguish between relevance and utility, and showed that documents annotated for utility by LLMs yield improved answer generation in retrieval-augmented generation (RAG) tasks compared to relevance-based annotations. Most existing work on LLM-based relevance annotation focuses primarily on building evaluation datasets [16], whereas research on the effectiveness of retrieval or ranking models trained using LLM annotations remains limited. Some studies have investigated the performance differences between training data generated by large language models and conventional training data [7, 8, 15]. Zhang et al. [51] compared models trained with LLM annotations versus human annotations for retrieval tasks, revealing the superiority of LLM annotation-supervised models in out-of-domain retrieval and RAG settings. However, there is still a lack of comprehensive comparison between models trained with LLM annotations and those trained with human annotations for ranking tasks, as well as limited investigation into how LLM annotations compare with other low-cost relevance supervision signals, such as click data.

Table 1: Annotation time and cost for different annotation strategies on the TianGong-ST dataset with ~260,000 query-document pairs.

Annotation Strategies	Time (h)	Cost ($)
PointAnn	52.0	332.8
ListAnn	7.6	48.5
ListRank	6.9	44.2
ListSel	6.1	38.8

2.3 Zero-shot Ranking with LLMs

In the ranking stage, particularly for re-ranking tasks, the candidate set is typically small, enabling multiple documents to be fed into the LLM simultaneously for direct ranking generation [27, 36, 55]. This approach requires the LLM to possess strong ranking capabilities, which smaller LLMs often lack. Consequently, many studies distill the ranking ability of large LLMs into smaller LLMs by fine-tuning them on permutations generated by the LLM [31, 32, 36]. However, generating full rankings from the LLM entails feeding multiple documents at once and, due to input length constraints, thus requires multiple passes to produce a global ranking. This results in high online deployment costs and substantial inference overhead. In contrast, distilling the LLM's relevance judgment into smaller models offers a more lightweight and flexible alternative.

3 LLM Annotations and Analysis

Given the high cost of human annotations, it is typically conducted by labeling each query-document pair individually according to predefined guidelines. Meanwhile, click signals generally provide only single-level supervision. In contrast, LLMs enable a highly flexible approach to relevance annotation. In this section, we describe how relevance annotations are generated using LLMs and present a relatively detailed analysis of the annotations.

3.1 Annotation Strategies

The flexibility of LLM-based relevance annotation is reflected in two main aspects: 1) the input format of query-document pairs, which can vary depending on the prompt design; and 2) the manner in which annotations are generated—LLMs can produce multi-level relevance annotations according to different annotation guidelines, or directly perform relevance selection or ranking over a list of candidate documents. As shown in Fig. 2, we employ three primary approaches to perform relevance annotation using LLMs:

- Multi-level annotation based on human-defined guidelines: LLMs generate graded relevance annotations following traditional annotation criteria. This approach can be further divided into two types based on the input format: **PointAnn**, single query-document pair input; and **ListAnn**, single query with multiple candidate documents as input.

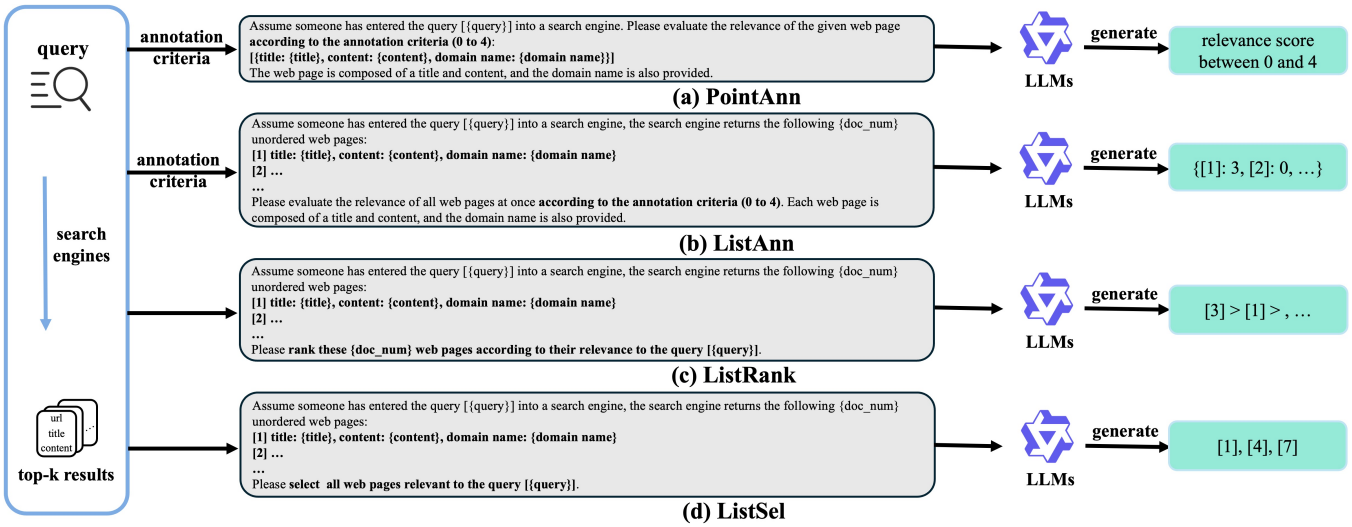


Figure 2: Annotation strategies with LLMs: (a) PointAnn, (b) ListAnn, (c) ListRank and (d) ListSel.

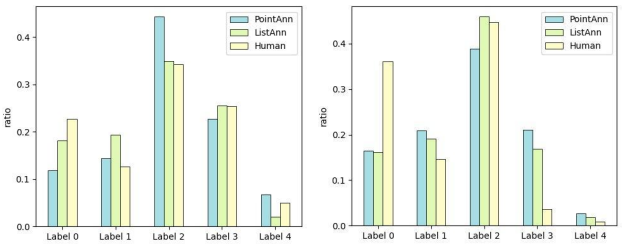
- Listwise Ranking (**ListRank**): The LLM is prompted to directly rank a list of candidate documents based on their relevance to the query.
- Listwise Selection (**ListSel**): The LLM is tasked with selecting a subset of documents from a given list that are relevant to the query.

Each of the three annotation strategies has its own strengths and weaknesses in terms of annotation accuracy, cost, and effectiveness in training ranking models. We utilize 8 × A800 GPU with a cost of \$0.8 per hour for a single A100 GPU[1], and employ Qwen2.5-32B-Instruct [38] for annotation. The annotation time and cost associated with these different annotation strategies are shown in the Tab. 1. Since we annotate the top ($\leq$ 10) results returned from search logs, all documents can be input to the LLMs at once. Although listwise input involves more content and generates more output, the overall annotation time remains short. Multi-level annotation provides fine-grained supervision but can be highly sensitive to the given annotation criteria. Ranking-based and selection-based annotations rely on comparisons among multiple documents, which are constrained by the requirement to present multiple documents at once for comparison. This is particularly problematic for ranking-based methods, where obtaining a global ranking requires multiple annotations. In addition, meaningful permutations depend on the LLM's ability to rank effectively. Nonetheless, it offers more fine-grained signals of relative relevance between documents, effectively leveraging such signals is critical. In contrast, selection-based annotations provide binary annotation signals, which may lack sufficient granularity to serve as strong supervision when used for model training.

3.2 Consistency with Human Annotations

Furthermore, we analyze the alignment between different annotation methods and human annotations. Considering that the ultimate goal is to train ranking models using annotated data, the relative relevance ranking within a document list is more important

[1]https://vast.ai/pricing/gpu/A800-PCIE



(a) TianGong-ST (b) Baidu-Click

Figure 3: Distribution of PointAnn, ListAnn, and human annotations at various relevance levels.

than individual annotations. Therefore, we primarily focus on the consistency of ranking orders between multi-level annotations and human-annotated rankings, as well as their ranking performance. We employ the Spearman correlation [20] and the Kendall correlation [1] to measure ranking consistency, and NDCG@10 [23] on the test set to evaluate ranking effectiveness.

Before conducting the consistency analysis, we first examine the distribution of LLM annotations, including PointAnn and ListAnn, as their formats are completely consistent with human annotations. This allows us to more directly perceive the characteristics of LLM annotations. As shown in Fig. 3, compared to human annotations, the proportion at label 0 of LLM annotations drops sharply, suggesting that LLMs impose a looser relevance threshold during annotation, which increases the risk of false positives. Here, label 2 corresponds to basic relevance, while labels 3 and 4 further consider timeliness and authority. On the Baidu-Click dataset, LLM annotations tend to more aggressively assign documents to labels 3 and 4, possibly because the returned results are generally of higher quality and LLMs inherently have limited ability to capture document-level signals, making them more easily misled. Considering that consistency metrics such as Kendall do not differentiate the importance of discordant pairs across different relevance levels, while NDCG@10 give greater weight on documents ranked higher, we

Table 2: Comparison of consistency metrics between different LLM annotation strategies and human annotations

Annotation Strategies	Ranking Consistency		Ranking Performance
	Spearman	Kendall	NDCG@10
PointAnn	0.2651	0.2509	0.8481
ListAnn	**0.3022**	**0.2853**	**0.8662**
ListRank	0.2534	0.2211	0.8577

calculate the consistency metrics focusing on documents with human annotations larger than 2.

As shown in Tab. 2, providing the input document list, the annotations obtained by giving explicit annotation guidelines are more consistent with human annotations and achieve better performance on the test set compared to directly letting the LLM perform ranking. Furthermore, given the annotation guidelines, compared to annotating documents sequentially for each query-document pair, providing the LLM with the complete document list and having it annotate all documents at once results in higher consistency and better ranking performance. Here, we only present the overall consistency results. The relative consistency levels of different annotation strategies across query frequencies remain consistent with the overall trends.

4 Can LLM Annotations Replace Clicks for LTR

In this section, we detail three comparison dimensions of clicks and LLM annotations for LTR, the training methods for LLM annotations and click data and the design of the ranking models. These investigations provide a comprehensive comparison between models trained with LLM annotations and those trained with click data, ultimately aiming to answer whether LLM annotations can replace clicks for LTR.

4.1 Comparison Dimensions for LTR

To answer the question of whether LLM annotations can replace user clicks for LTR, we first analyze the relevance judgment capabilities of each supervision. The relevance inferred from user clicks reflects not only query-document semantic matching, but also document-level signals such as authority, and content quality. In contrast, LLMs possess strong semantic understanding capabilities but are limited to perceiving document-level signals—for example, it is challenging for LLMs to infer authority or quality from a url's domain name. Therefore, LLM annotations are particularly advantageous in capturing semantic matching between queries and documents.

Given that search engines tend to perform differently across queries of varying frequency—typically achieving better performance on high-frequency queries—the results list under high-frequency queries usually contains a higher proportion of relevant documents, making user clicks more influenced by document-level factors (e.g., authority or quality). Thus, we conduct a fine-grained comparison across queries of different frequencies to analyze how click data and LLM annotation impact ranking performance.

Secondly, given the different focuses of clicks and LLM annotations, we incorporate query-document matching features (e.g., BM25 [35]) and document-level features (e.g., PageRank [30]) besides semantic embeddings, aiming to further examine the capabilities of models trained with each annotation and how LTR features from different levels complement them.

Since users typically do not examine results beyond the first page, click data is generally collected from only the top 10 results [56]. Moreover, truly irrelevant documents are rare among these top results, making it difficult to evaluate how well a model distinguishes relevance from irrelevance. To investigate this capability, we introduce truly irrelevant documents into the test set (under the condition that the training and test sets are from the same distribution). Additionally, we further explore the impact of incorporating truly irrelevant examples into the training set on ranking performance.

4.2 Training Methods

In this section, we introduce the specific training methods employed for different LLM-based annotation approaches, as well as various ULTR methods for click data.

For a given query q and its documents list D, the loss function calculated for a document d d is denoted as $\mathcal{L}$. We use l_d and c_d to represent the LLM annotation and click label of d, respectively. We denote the ranking model as f.

4.2.1 LLM Annotation-Supervised Training. For PointAnn, ListAnn, and ListRel, we adopt the listwise loss for training [3]:

$$\mathcal{L}_{list} = -l_d \cdot log\frac{e^{f(q,d)}}{\sum_{d' \in D} e^{f(q,d')}}. \tag{1}$$

For ListRank, we employ two types of training methods: (1) use $10 - \text{rank}_d$ as the relevance label and training with the listwise loss, where rank_d represents the rank of d in the permutation; (2) training with the RankNet loss [9]:

$$\mathcal{L}_{pair} = \sum_{d_i \in D} \sum_{d_j \in D} \mathrm{I}(l_{d_i} < l_{d_j}) \cdot log(1 + exp(f(q, d_i) - f(q, d_j))). \tag{2}$$

RankNet loss is a pairwise loss that optimizes the relative relevance between pairs of documents. The ranking results produced by the LLM generate $\frac{n(n-1)}{2}$ document pairs.

4.2.2 Unbiased Learning to Rank. Unbiased Learning to Rank (ULTR) methods are mostly based on the user examination hypothesis [34]:

$$P(c_d = 1) = P(r_d = 1) \cdot P(o_d = 1), \tag{3}$$

which states that a document d id clicked ($c_d = 1$) if and only if it is relevant ($r_d = 1$) and observed by the user ($o_d = 1$). To obtain an unbiased ranking model f, the key lies in accurately estimating the propensity model g, i.e., $P(o_d = 1)$. Note that we uniformly utilize the listwise cross-entropy loss function with inverse propensity weighting (IPW) [24, 46] for training:

$$\mathcal{L}_{IPW} = \mathcal{L}(f(q, d); \frac{c_d}{g}). \tag{4}$$

Many existing studies primarily focus on mitigating position bias, assuming that the probability of a document being observed depends solely on its position. However, real click data arise from complex underlying user interactions, making the biases more intricate and not limited to position bias alone. On the one hand, the

observation probability of a document depends not only on its position but also on user behavior related to surrounding documents. On the other hand, due to the stronger performance of ranking models in real-world scenarios, there exists a strong coupling between position and relevance.

Therefore, we adopt the following ULTR methods:

- **Naive**: is a commonly used ULTR method, which directly uses clicks as relevance, without additional debiasing:

$$\mathcal{L}_{Naive} = \mathcal{L}(f(q,d); c_d). \tag{5}$$

- **Dual Learning Algorithm (DLA)** [4]: is a commonly used method to mitigate position bias, simultaneously training an unbiased ranking model and an unbiased propensity model:

$$\mathcal{L}_{DLA} = \mathcal{L}(f(q,d); \frac{g_1}{g_{k_d}} c_d) + \mathcal{L}(g(k_d); \frac{f(q,d_1)}{f(q,d)} c_d), \tag{6}$$

  where k_d represents the position of the document d in the documents list.
- **Interactional Observation-Based Model (IOBM)** [12]: compared to DLA, IOBM considers positions and clicks of both the document itself and the surrounding documents to model the propensity model:

$$\mathcal{L}_{IOBM} = \mathcal{L}(f(q,d); \frac{g_1}{g_{k_d}} c_d) + \mathcal{L}(g(k_d; \mathbf{c_q}); \frac{f(q,d_1)}{f(q,d)} c_d), \tag{7}$$

  where c_q represents the click information of the document list under query
- **Unconfounded Propensity Estimation (UPE)** [26]: leverages backdoor adjustment to address propensity overestimation:

$$\mathcal{L}_{UPE} = \mathcal{L}(f(q,d); \frac{g_{do(k_1)}}{g_{do(k_d)}} c_d) + \mathcal{L}(g(k_d); \frac{f(q,d_1)}{f(q,d)} c_d). \tag{8}$$

- **Observation Dropout (Drop)** and **Gradient Reversal (GradRev)** [53]: these two methods aim to disentangle observation and relevance. **Drop** introduces a dropout layer following the propensity model, whereas **Gradrev** appends a negative gradient reversal layer after the propensity model, where the negative gradient originates from the fitting loss $\mathcal{L}'_{rev}$ of the propensity model with respect to relevance:

$$\begin{aligned} \mathcal{L}_{Drop} &= \mathcal{L}(f(q,d); \frac{g_1}{g_{k_d}} c_d) + \mathcal{L}(Dropout(g(k_d)); \frac{f(q,d_1)}{f(q,d)} c_d), \\ \mathcal{L}_{Gradrev} &= \mathcal{L}(f(q,d); \frac{g_1}{g_{k_d}} c_d) + \mathcal{L}(g(k_d); \frac{f(q,d_1)}{f(q,d)} c_d) + \mathcal{L}'_{rev}. \end{aligned} \tag{9}$$

4.3 Ranking Models

State-of-the-art ranking models commonly adopt a cross-encoder architecture [6], where the input is concatenated as "[CLS] query [SEP] document [SEP]". The [CLS] token representation is extracted and used to predict the relevance between the query and the document. We choose ERNIE-3.0 Base [37] as the backbone due to its strong semantic representation capabilities for Chinese text.

To further compare the capabilities of models trained with the two types of annotation and to examine how LTR features from different levels can complement these models, we incorporate such features and concatenate them with the relevance scores extracted from the fine-tuned cross-encoder models. The combined representations are then used to train a lightweight Deep Neural Network (DNN)-based ranking model. We do not employ tree-based models here, as debiasing methods for real click data have shown limited effectiveness when applied to tree-based approaches [21, 56]. The extracted features are shown in Tab. 3.

5 Experimental Setup

Dataset. We use both a public click dataset, TianGong-ST [11], and a private dataset, Baidu-Click. **TianGong-ST** consists of 18 days of search logs collected from the Chinese search engine, Sogou. Each query is associated with the top-10 displayed results and corresponding click feedback. A subset of 2k queries sampled from the logs have human annotations ranging from 0 to 4. After filtering, there are about 260k queries with at least one clicked document in the top-10 list, which are used for training. **Baidu-Click** is collected from Baidu, the largest Chinese search engine. Here, we also sample about 260k queries from a week's search logs, each associated with the top-7 organic results and their click information. From the subsequent week, we sample about 7k queries whose top-50 ranking-stage results are annotated with 5-level human annotations (0–4). For both datasets, we split the human annotations into validation and test sets with a 3:7 ratio. Note that for the Baidu-Click dataset, we adopt its pre-defined query frequency categories and group queries into three coarse-grained bins: high, medium, and low frequency. For TianGong-ST, we compute query frequencies from search logs and apply equal-frequency binning to partition queries into the same three categories. In addition, we extract LTR features for both datasets, as shown in Tab. 3.

Metrics. To evaluate ranking performance, we report nDCG [23] at positions 1, 3, 5, and 10. We further report these metrics separately for queries of different frequency categories. All experimental results are averaged over three random seeds.

Implementation Details. Since the document content in TianGong-ST is crawled from original web pages, it contains substantial irrelevant information (e.g., navigation bars) unrelated to the main content. To address this, we employ DeepSeek-R1-Distill-Qwen-32B to extract the core content, effectively generating a document summary. When fine-tuning ERNIE-3.0 Base, we set the maximum sequence length to 160 and 640 for TianGong-ST and Baidu-Click, respectively.

The [CLS] representation extracted from ERNIE-3.0 Base is fed into a three-hidden-layer deep neural network (DNN) with hidden sizes of [512, 256, 128] to produce the final relevance score. For the ranking model that takes the concatenation of LTR features and the relevance score extracted from the fine-tuned ERNIE-3.0 Base as input, the concatenated features are first mapped to a 64-dimensional vector, which is then passed through a three-hidden-layer DNN with hidden sizes of [32, 16, 8] to generate the final relevance score. Each hidden layer is followed by an ELU activation function.

We implement the ULTR methods described in Sec. 4.2.2 using the ULTRA toolkit [40]. For each ULTR method, we keep all model-specific parameters consistent with their original papers. We set the batch size to 64, use the AdamW optimizer with a learning rate of 3e-5, and apply early stopping if no improvement is observed on the validation set for 3 consecutive epochs. The maximum number of training epochs is set to 8.

Table 3: Overview of different-level LTR features from real click data.

Level	Feature Name	Feature Description
Query-Document	TF, IDF, TF·IDF	The sum of term frequency (TF), inverse document frequency (IDF) and TF·IDF of query terms in title, content and the whole document.
	BM25 [35]	The scores of BM25 on title, content and the whole document.
	LMJM, LMDIR, LMABS [49]	The scores of launguage model (LM) with different smoothing methods—Jelinek-Mercer (JM), Dirichlet (DIR), and absolute discounting (ABS)—on title, content, and whole document.
	proximity1 [10]	Averaged proximity score and positions of query terms within the whole document, computed both including and excluding stop words.
	proximity2 [10]	Number of query bigrams appearing in the whole document, computed both including and excluding stop words, within a window of 5 and 10 words.
Document	Length	The length of title, content, url and the whole document.
	Slash	The number of slash in url.
	PageRank [30]	The PageRank score of url domain.

6 Results and Analysis

6.1 Comparative Training Performance

Tab. 4 presents the overall and frequency-specific performance of various ULTR methods and LLM annotation-based training approaches. We first separately compare the training approaches within each annotation type. When trained on real-world click data, the performance gap between DLA and the Naive baseline is minimal, indicating that addressing only position bias is insufficient for real-world click data. Overall, the GradRev method achieves the best performance.

When trained on LLM annotations, models trained with multi-level annotations (PointAnn, ListAnn and ListRank) outperform those trained with single-level annotations (ListSel); among multi-level methods, PointAnn yields the worst performance, highlighting the advantage of listwise input strategies. Meanwhile, the two training approaches based on the ListRank annotation strategy perform similarly but are inferior to ListAnn, possibly due to performance degradation caused by overemphasizing individual document relevance when the generated ranking quality is suboptimal. The model performance across different annotation strategies with LLMs largely aligns with their agreement with human annotations in Tab. 2, indicating that the consistency with human annotations can serve as a reliable criterion for selecting annotation strategies.

We then compare the performance of models trained on click data and LLM annotations. Overall, models supervised by LLM annotations outperform those supervised by clicks. However, considering query frequencies, LLM annotation-supervised models significantly excel on medium- and low-frequency queries, while click-supervised models retain an advantage on high-frequency queries. This suggests LLM annotations cannot yet fully replace clicks. The fundamental reason for this performance difference derives from the differences in their abilities to capture relevance. In real-world scenarios, ranking models generally perform better on high-frequency queries, returning more relevant results, thus document-level signals like authority and quality become crucial for distinguishing relevance. Click data can capture these signals, whereas LLM annotations struggle to do so. Attempts to incorporate numeric signals like PageRank into LLM annotations were unsuccessful, likely due to LLMs' limited ability to interpret numerical data. Conversely, medium- and low-frequency queries rely more on semantic matching, highlighting the strength of LLM annotations in capturing semantic relevance.

In subsequent experiments, we consistently use the GradRev method on click data and the ListAnn annotation strategy for training. We validate the performance differences between the two on the Baidu-Click dataset. As shown in the Tab. 4, models trained with click data maintain their advantage on high-frequency queries, while models trained with LLM annotations perform better on medium- and low-frequency queries, which is consistent with the conclusions above.

6.2 Impact of LTR Feature Groups

We extract relevance scores from models trained on both click data and LLM annotations. For each type of supervision, these relevance scores are concatenated with LTR features to train a lightweight ranking model.

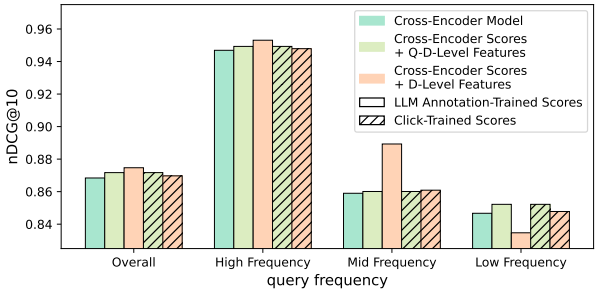
For click data, as shown in Fig. 4(a), concatenating document-level (d-level) features with relevance scores from the model trained on LLM annotations yields greater performance improvements on high- and medium-frequency queries compared to query-document-level (q-d level) features. Conversely, when concatenated with relevance scores from the click-trained model, q-d level features bring more performance gains than d-level features.

For LLM annotations, as shown in Fig. 4(b), concatenating d-level features with relevance scores from the LLM annotation-trained model improves performance on high- and medium-frequency queries, while q-d level features enhance performance on low-frequency queries. When concatenated with relevance scores from the click-trained model, q-d level features improve performance on high-frequency queries.

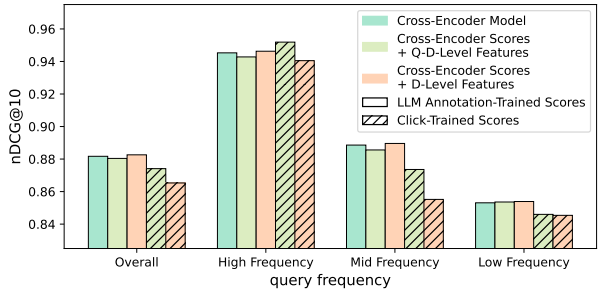
Combining these observations with the differences in relevance judgment capabilities between click data and LLM annotations, we conclude the following: under the cross-encoder architecture, 1)

Table 4: Performance comparison of various methods trained with click data and LLM annotations. Bold and underline indicate the best and second-best methods within the same type of annotation; for the Gradrev method, * indicates better performance than the model trained on LLM annotations.

Datasets	Methods	nDCG@1				nDCG@3				nDCG@5				nDCG@10			
		All	High	Mid	Low	All	High	Mid	Low	All	High	Mid	Low	All	High	Mid	Low
TianGong-ST	Naive	0.6992	0.9224	0.6849	0.6296	0.7098	0.8659	0.6930	0.6680	0.7389	0.8645	0.7229	0.7078	0.8673	0.9433	**0.8597**	0.8452
	DLA	0.6990	0.9224	0.6746	0.6395	0.7068	0.8785	0.6823	0.6669	0.7367	0.8613	0.7184	0.7082	0.8660	0.9437	0.8567	0.8447
	IOBM	0.6938	0.9224	0.6596	**0.6423**	0.7042	0.8753	0.6670	**0.6773**	0.7359	0.8646	0.7129	**0.7106**	0.8652	0.9429	0.8504	**0.8480**
	UPE	**0.7246**	0.9230	**0.7330**	0.6412	0.7134	0.8884	**0.6963**	0.6648	0.7391	0.8664	**0.7313**	0.6991	0.8664	0.9459	0.8567	0.8445
	Drop	0.7067	0.9224	0.6969	0.6352	0.7116	0.8742	0.6944	0.6677	**0.7408**	0.8621	0.7269	0.7092	0.8668	0.9425	0.8587	0.8452
	Gradrev	0.7013	**0.9232***	0.6789	0.6403*	**0.7150**	**0.8926***	0.6949	0.6683	0.7399	**0.8710**	0.7210	0.7097	**0.8684**	**0.9469***	0.8590	0.8467
	PointAnn	0.6830	0.6950	0.7393	0.6214	0.7234	0.7474	0.7573	0.6801	0.7615	0.7865	0.7824	0.7309	0.8643	0.8690	0.8806	0.8502
	ListAnn	**0.7165**	**0.8332**	**0.7782**	0.6101	**0.7570**	**0.8690**	0.7802	0.6913	**0.7869**	**0.8837**	0.7999	0.7373	**0.8817**	**0.9453**	**0.8886**	0.8531
	ListRank_list	0.6613	0.5227	0.7459	0.6278	0.7407	0.7559	0.7780	0.6972	0.7675	0.7665	0.7941	0.7409	0.8632	0.8553	0.8783	0.8546
	ListRank_pair	0.6704	0.4996	0.7657	**0.6382**	0.7447	0.7471	**0.7873**	**0.7006**	0.7715	0.7604	**0.8020**	**0.7448**	0.8664	0.8508	0.8851	**0.8578**
	ListSel	0.5820	0.4561	0.6470	0.5635	0.6319	0.5575	0.6567	0.6347	0.6993	0.6788	0.7173	0.6889	0.8179	0.7849	0.8231	0.8261
	Zero-shot	0.6442	0.5833	0.7036	0.6070	0.7002	0.7117	0.7235	0.6723	0.7455	0.7517	0.7606	0.7278	0.8579	0.8464	0.8714	0.8486
Baidu-Click	Gradrev	0.5460	0.5465*	0.5749	0.5276	0.5873	0.5831*	0.6148	0.5715	0.5920	0.5954*	0.6253	0.5675	0.6057	0.6270*	0.6383	0.5714
	ListAnn	0.5781	0.5401	0.5946	0.5849	0.6147	0.5779	0.6330	0.6202	0.6171	0.5921	0.6395	0.6147	0.6311	0.6201	0.6540	0.6214

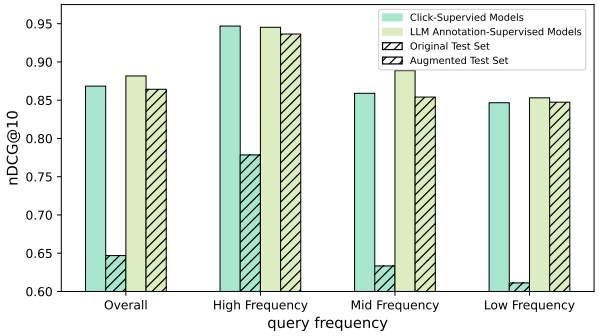


(a) Models Trained with Click Data

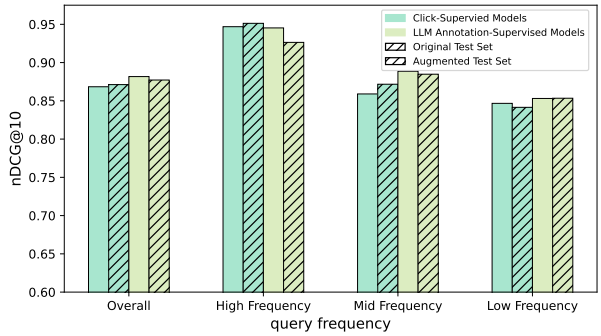


(b) Models Trained with LLM Annotations

Figure 4: Effects of document-level and query-document-level LTR features on models.



(a) Incorporating True Negatives in the Test Set



(b) Incorporating True Negatives in the Train Set

Figure 5: Effects of true negatives in training and test sets on model performance.

models trained on click data capture both query-document semantic matching and document-level signals such as authority, and the introduction of q-d level features can further enhance their capability; 2) models trained on LLM annotations excel at capturing query-document semantic matching and show weak ability to capture document-level signals. This may be attributed to the listwise input annotation strategy, where the LLM can implicitly detect subtle document-level differences. Consequently, incorporating d-level features can modestly improve the performance of LLM annotation-trained models.

6.3 Exploring Relevance Discrimination via True Negatives

Since the training set contains a large number of relevant results, the trained ranking models are able to distinguish which relevant documents are more relevant. To investigate whether the models can truly differentiate between relevant and non-relevant documents, we introduced 10 true negative samples into the original test set on TianGong-ST (since its test set follows the same distribution as the training set). As shown in Fig. 5(a), after adding true negatives to the test set, the performance of models trained with both annotation types declined. However, the performance drop of click-supervised models is significantly larger than that of LLM annotation-supervised models. This indicates that models trained with LLM annotations not only distinguish relevance differences within relevant results but also effectively separate relevant from non-relevant documents, whereas click-supervised models have weaker discrimination ability between relevant and non-relevant documents.

Furthermore, we introduce 10 true in-batch negatives into the training set to enhance the models' ability to differentiate relevant from non-relevant documents and explore the impact on both types of trained models. As illustrated in Fig. 5(b), for click-supervised models, incorporating true negatives in training further improved ranking performance by strengthening relevance discrimination. In contrast, since LLM annotation-supervised models already possess this ability, adding true negatives caused the models to focus more on the obvious boundary between relevant and non-relevant documents, which in turn led to an overall performance decline, as the models require more nuanced discrimination within relevant documents.

7 Hybrid Training: Integrating Clicks and LLM Annotations

The above experimental results indicate that under the cross-encoder architecture, the relevance discrimination capabilities of models trained on click data and those trained on LLM annotations align closely with the inherent strengths of their respective supervision signals. This alignment manifests as performance advantages on queries of different frequencies, highlighting a natural complementarity between the two types of annotation. These findings suggest that queries with varying frequencies may benefit from distinct supervision signals.

Motivated by this, we further investigate effective strategies to integrate click data and LLM annotations within the cross-encoder framework, aiming to train ranking models that possess strong capabilities in capturing both semantic matching signals and document-level information simultaneously. In this section, we provide a detailed description and experimental results of the two integration approaches we propose: Data Scheduling and Frequency-Aware Multi-Objective Learning.

7.1 Data Scheduling for Hybrid Supervision

Motivated by the following two considerations: 1) LLM annotations exhibit strong capability in capturing semantic matching between queries and documents, whereas click data captures both semantic matching and document-side signals such as authority; 2) although

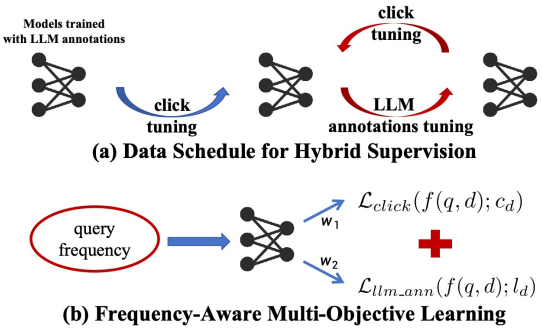


Figure 6: Hybrid training methods: (a) Data Schedule and (b) Frequency-Aware Multi-Objective Learning.

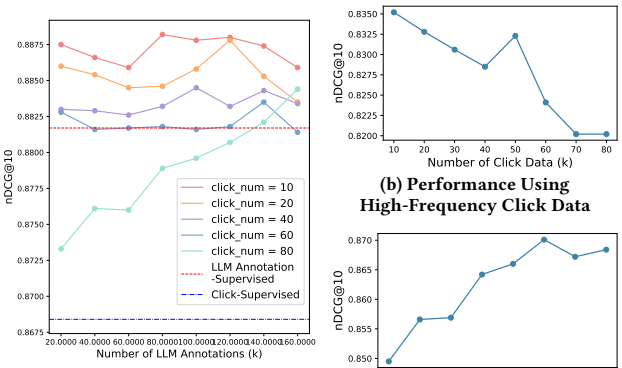
obtaining annotations via even open-source LLMs is still relatively more expensive compared to click data, their annotation quality is superior.

We first train the ranking model on LLM annotations to establish a solid foundation in semantic matching-based relevance judgment. To further enhance the model's ability to discriminate relevance by incorporating document-level signals, we continue training the model with click data. However, since the model may suffer from catastrophic forgetting during training, to mitigate this, we adopt an alternating training strategy that iteratively trains the ranking model on click data and LLM annotations. Considering the experimental observation that click-trained models perform better on high-frequency queries while LLM annotation-trained models excel on medium- and low-frequency queries, we use high-frequency queries for click data and medium- and low-frequency queries for LLM annotations during alternating training, as shown in Fig. 6(a).

We explore two alternating schemes: fine-grained mixing and coarse-grained alternation. Fine-grained mixing randomly mixes click data and LLM annotations within each training batch, whereas coarse-grained alternation trains separately on one annotation type at a time and switches periodically. Our experiments show that fine-grained mixing does not improve performance, possibly due to conflicting training objectives causing model confusion. For coarse-grained alternation, the ratio of click to LLM data used in alternation becomes a crucial factor.

As shown in Fig. 7(a), in the data scheduling method, integrating only 10k of click data results in an overall performance improvement (with improvements also observed across different frequency levels). Moreover, increasing the amount of LLM annotations does not necessarily lead to better results. With 10k of click data, the overall performance consistently outperforms the original warm-start model trained on LLM annotations. This suggests that training with 10k of data helps improve the model's weaker capabilities while also preserving or even enhancing its original strengths.

Furthermore, we separately train ranking models using click data for high-frequency queries and LLM annotations for mid- and low-frequency queries, as shown in Fig. 7(b). We find that when training solely with click data for high-frequency queries, the optimal performance is achieved with a data volume of 10k. It indicates that, to achieve optimal performance in the data scheduling, the required amount of click data is likely consistent with the optimal amount for training alone. Similarly, the amount of LLM annotations required is also related to its optimal amount for separate training (120k).



(a) Performance with Varying Amounts of Click Data and LLM Annotations

(b) Performance Using High-Frequency Click Data

(c) Performance Using Mid- and Low-Frequency LLM Annotations

Figure 7: Impact of Varying Volumes of Click and LLM Annotations on Data Scheduling.

Table 5: Performance comparison (nDCG@10) of methods combining click data and LLM annotations. Bold indicates the best-performing method within the same supervision type, while * indicates that a hybrid training method outperforms even the best single-supervision method.

Methods	TianGong-ST				Baidu-Click			
	All	High	Mid	Low	All	High	Mid	Low
Single Supervision								
Gradrev	0.8684	**0.9469**	0.8590	0.8467	0.6057	**0.6270**	0.6383	0.5714
ListAnn	**0.8817**	0.9453	**0.8886**	**0.8531**	**0.6311**	0.6201	**0.6540**	**0.6214**
Hybrid Supervision								
Schedule	0.8878*	**0.9592***	**0.8952***	0.8560*	-	-	-	-
FAMOL	**0.8897***	0.9533*	0.8917*	**0.8649***	0.6333*	0.6275*	0.6589*	0.6200

7.2 Frequency-Aware Multi-Objective Learning

Queries of different frequencies require different types of supervision signals. Based on this insight, we design a frequency-aware weighting mechanism formalized as Eq. 10 and is shown in Fig. 6(b).

$$w_1, w_2 = sigmoid(\text{FFN}(\text{Emb}(q_{freq}))),$$
$$\mathcal{L} = w_1 \cdot \mathcal{L}_{click}(f(q,d); c_d) + w_2 \cdot \mathcal{L}_{llm\_ann}(f(q,d); l_d). \quad (10)$$

Let q_{freq} denote the query frequency. We set the embedding dimension to 16, and FFN is a two-layer DNN with sizes [64, 2].

As shown in Tab. 5, compared to models trained solely on a single supervision signal, the frequency-aware weighting mechanism adaptively adjusts the training weights of the two supervision signals based on the current query frequency, thereby leveraging the advantages of both.

Both data scheduling (Schedule) and frequency-aware multi-objective learning (FAMOL) methods improve performance across queries of all frequency levels as shown in Tab. 5, indicating that both approaches can effectively leverage the strengths of the two types of annotations simultaneously. Among the two approaches, FAMOL demonstrates more consistent performance improvements. Overall, we validate the effectiveness of FAMOL on the Baidu-Click dataset.

8 Conclusion and Future Work

This paper investigates whether LLM annotations can replace click data for learning to rank (LTR). Through comprehensive comparisons, we show that while LLM annotations capture semantic relevance well, click data additionally reflect document-level signals such as authority. Experiments across queries of varying frequencies reveal that click-supervised models perform better on high-frequency queries, whereas LLM annotation-supervised models excel on medium- and low-frequency ones. We further introduce semantic matching and document-level features to analyze how different levels of LTR features complement each annotation. Results suggest that semantic features better support click-based models, while document-level features may benefit LLM annotation-based ones. Finally, we explore hybrid training strategies—data scheduling and frequency-aware multi-objective learning—which effectively combine the strengths of both annotations, with the latter showing superior performance.

At the current stage, where both click data and LLM annotations are available, our work provides guidance on how to leverage these two types of annotation to train ranking models with improved performance. We propose two strategies to combine the two types of annotations, yet finding more effective ways to integrate them remains an open question.

9 Ethical Statement

In this work, the public and private click data we use contain no real user information and therefore do not infringe on user privacy. And we will make our code publicly available soon. We investigate the impact of training ranking models using click data and LLM annotations, aiming to answer whether LLM annotations can replace click data. Currently, these two annotation methods are generated independently. However, if in the future LLM annotations are used for personalized search, providing certain user privacy information to the LLM to generate personalized annotations could raise privacy concerns. Therefore, when exploring LLM-based personalized search in the future, it will be crucial to safeguard privacy boundaries.

References

[1] Hervé Abdi. 2007. The Kendall rank correlation coefficient. *Encyclopedia of measurement and statistics* 2 (2007), 508–510.

[2] Aman Agarwal, Xuanhui Wang, Cheng Li, Michael Bendersky, and Marc Najork. 2019. Addressing trust bias for unbiased learning-to-rank. In *The World Wide Web Conference*. 4–14.

[3] Qingyao Ai, Keping Bi, Jiafeng Guo, and W Bruce Croft. 2018. Learning a deep listwise context model for ranking refinement. In *The 41st international ACM SIGIR conference on research & development in information retrieval*. 135–144.

[4] Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and W Bruce Croft. 2018. Unbiased learning to rank with unbiased propensity estimation. In *The 41st international ACM SIGIR conference on research & development in information retrieval*. 385–394.

[5] 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.

[6] Shivaji Alaparthi and Manit Mishra. 2020. Bidirectional Encoder Representations from Transformers (BERT): A sentiment analysis odyssey. *arXiv preprint arXiv:2007.01127* (2020).

[7] Arian Askari, Mohammad Aliannejadi, Evangelos Kanoulas, and Suzan Verberne. 2023. A test collection of synthetic documents for training rankers: Chatgpt vs. human experts. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 5311–5315.

[8] Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Data augmentation for information retrieval using large language models. *arXiv preprint arXiv:2202.05144* (2022).

[9] Christopher JC Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning* 11, 23-581 (2010), 81.

[10] Jia Chen, Haitao Li, Weihang Su, Qingyao Ai, and Yiqun Liu. 2023. Thuir at wsdm cup 2023 task 1: Unbiased learning to rank. *arXiv preprint arXiv:2304.12650* (2023).

[11] Jia Chen, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2019. TianGong-ST: A new dataset with large-scale refined real-world web search sessions. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2485–2488.

[12] Mouxiang Chen, Chenghao Liu, Jianling Sun, and Steven CH Hoi. 2021. Adapting interactional observation embedding for counterfactual learning to rank. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*. 285–294.

[13] Aleksandr Chuklin, Ilya Markov, and Maarten De Rijke. 2022. *Click models for web search*. Springer Nature.

[14] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An experimental comparison of click position-bias models. In *Proceedings of the 2008 international conference on web search and data mining*. 87–94.

[15] Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B Hall, and Ming-Wei Chang. 2022. Promptagator: Few-shot dense retrieval from 8 examples. *arXiv preprint arXiv:2209.11755* (2022).

[16] Mouly Dewan, Jiqun Liu, and Chirag Shah. 2025. LLM-driven usefulness labeling for IR evaluation. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 3055–3059.

[17] Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. ChatGPT outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences* 120, 30 (2023), e2305016120.

[18] Shashank Gupta, Philipp Hager, Jin Huang, Ali Vardasbi, and Harrie Oosterhuis. 2024. Unbiased Learning to Rank: On Recent Advances and Practical Applications. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. 1118–1121.

[19] Philipp Hager, Romain Deffayet, Jean-Michel Renders, Onno Zoeter, and Maarten de Rijke. 2024. Unbiased learning to rank meets reality: Lessons from Baidu's large-scale search dataset. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1546–1556.

[20] Jan Hauke and Tomasz Kossowski. 2011. Comparison of values of Pearson's and Spearman's correlation coefficients on the same sets of data. *Quaestiones geographicae* 30, 2 (2011), 87–93.

[21] Ziniu Hu, Yang Wang, Qu Peng, and Hang Li. 2019. Unbiased lambdamart: an unbiased pairwise learning-to-rank algorithm. In *The World Wide Web Conference*. 2830–2836.

[22] Fan Huang, Haewoon Kwak, and Jisun An. 2023. Is chatgpt better than human annotators? potential and limitations of chatgpt in explaining implicit hate speech. In *Companion proceedings of the ACM web conference 2023*. 294–297.

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

[24] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of the tenth ACM international conference on web search and data mining*. 781–789.

[25] Taja Kuzman, Igor Mozetič, and Nikola Ljubešić. 2023. Chatgpt: Beginning of an end of manual linguistic data annotation? use case of automatic genre identification. *arXiv preprint arXiv:2303.03953* (2023).

[26] Dan Luo, Lixin Zou, Qingyao Ai, Zhiyu Chen, Chenliang Li, Dawei Yin, and Brian D Davison. 2023. Unconfounded Propensity Estimation for Unbiased Ranking. *arXiv preprint arXiv:2305.09918* (2023).

[27] Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. 2023. Zero-shot listwise document reranking with a large language model. *arXiv preprint arXiv:2305.02156* (2023).

[28] Ben Mann, Nick Ryder, Melanie Subbiah, J Kaplan, P Dhariwal, A Neelakantan, P Shyam, G Sastry, A Askell, S Agarwal, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* 1, 3 (2020), 3.

[29] Zohreh Ovaisi, Ragib Ahsan, Yifan Zhang, Kathryn Vasilaky, and Elena Zheleva. 2020. Correcting for selection bias in learning-to-rank systems. In *Proceedings of the web conference 2020*. 1863–1873.

[30] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web*. Technical Report. Stanford infolab.

[31] Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023. Rankvicuna: Zero-shot listwise document reranking with open-source large language models. *arXiv preprint arXiv:2309.15088* (2023).

[32] Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023. RankZephyr: Effective and Robust Zero-Shot Listwise Reranking is a Breeze! *arXiv preprint arXiv:2312.02724* (2023).

[33] Hossein A Rahmani, Clemencia Siro, Mohammad Aliannejadi, Nick Craswell, Charles LA Clarke, Guglielmo Faggioli, Bhaskar Mitra, Paul Thomas, and Emine Yilmaz. 2025. Judging the judges: A collection of llm-generated relevance judgements. *arXiv preprint arXiv:2502.13908* (2025).

[34] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks: estimating the click-through rate for new ads. In *Proceedings of the 16th international conference on World Wide Web*. 521–530.

[35] Stephen E Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR'94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, organised by Dublin City University*. Springer, 232–241.

[36] Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT good at search? investigating large language models as re-ranking agents. *arXiv preprint arXiv:2304.09542* (2023).

[37] Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. 2021. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137* (2021).

[38] Qwen Team. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671* (2024).

[39] Paul Thomas, Seth Spielman, Nick Craswell, and Bhaskar Mitra. 2024. Large language models can accurately predict searcher preferences. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1930–1940.

[40] Anh Tran, Tao Yang, and Qingyao Ai. 2021. ULTRA: an unbiased learning to rank algorithm toolbox. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4613–4622.

[41] Ali Vardasbi, Maarten de Rijke, and Ilya Markov. 2020. Cascade model-based propensity estimation for counterfactual learning to rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2089–2092.

[42] Ali Vardasbi, Maarten de Rijke, and Ilya Markov. 2021. Mixture-based correction for position and trust bias in counterfactual learning to rank. In *Proceedings of the 30th ACM international conference on information & knowledge management*. 1869–1878.

[43] Ali Vardasbi, Harrie Oosterhuis, and Maarten de Rijke. 2020. When inverse propensity scoring does not work: Affine corrections for unbiased learning to rank. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1475–1484.

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[45] 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.

[46] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position bias estimation for unbiased learning to rank in personal search. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 610–618.

[47] Andrew Yates, Rodrigo Nogueira, and Jimmy Lin. 2021. Pretrained transformers for text ranking: BERT and beyond. In *Proceedings of the 14th ACM International Conference on web search and data mining*. 1154–1156.

[48] Lulu Yu, Keping Bi, Jiafeng Guo, Shihao Liu, Dawei Yin, and Xueqi Cheng. 2025. Unbiased Learning to Rank with Query-Level Click Propensity Estimation: Beyond Pointwise Observation and Relevance. In *Companion Proceedings of the ACM on Web Conference 2025*. 1495–1499.

[49] Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS)* 22, 2 (2004), 179–214.

[50] Chengxiang Zhai and John Lafferty. 2017. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Acm sigir forum*, Vol. 51. ACM New York, NY, USA, 268–276.

[51] Hengran Zhang, Minghao Tang, Keping Bi, Jiafeng Guo, Shihao Liu, Daiting Shi, Dawei Yin, and Xueqi Cheng. 2025. Leveraging LLMs for Utility-Focused Annotation: Reducing Manual Effort for Retrieval and RAG. *arXiv preprint arXiv:2504.05220* (2025).

[52] Hengran Zhang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. 2024. Are Large Language Models Good at Utility Judgments?. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1941–1951.

[53] 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.

[54] Haiyuan Zhao, Jun Xu, Xiao Zhang, Guohao Cai, Zhenhua Dong, and Ji-Rong Wen. 2023. Unbiased Top-k Learning to Rank with Causal Likelihood Decomposition. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*. 129–138.

[55] Shengyao Zhuang, Honglei Zhuang, Bevan Koopman, and Guido Zuccon. 2024. A setwise approach for effective and highly efficient zero-shot ranking with large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 38–47.

[56] 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.