Check for updates

RankExpert: A Mixture of Textual-and-Behavioral Experts for Multi-Objective Learning-to-Rank in Web Search

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

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

Yongqi Zhang*
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, China
yongqizhang@hkust-gz.edu.cn

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

Mingxin Cai
Shanghai Jiao Tong University.
Shanghai, China
021033910047@sjtu.edu.cn

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

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

Linghe Kong
Shanghai Jiao Tong University
Shanghai, China
linghe.kong@sjtu.edu.cn

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

Lei Chen
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, China
leichen@cse.ust.hk

Abstract

As modern learning-to-rank (LTR) systems rely on both textual and behavioral features, it is essential to extend pre-trained language models (PLMs) from text (queries and webpages) understanding to end-to-end ranking score prediction subject to multiple objectives, such as relevance, quality, authority, and recency. While textual inputs encompass a broader array of features than mere relevance and behavioral features are frequently skewed by user feedback with position bias, an integrated solution is required to jointly disentangle and fuse these heterogeneous features, ensuring robust and unbiased ranking predictions. In this work, we introduce **RankExpert**, a unified framework that holistically models heterogeneous ranking signals by integrating PLM-based semantic extraction with behavioral cues. RankExpert employs a lightweight PLM with hierarchical distillation for efficient query-document representation, a mixture-of-experts (MoE) architecture to disentangle and optimize objectives across relevance, quality, authority, and recency, and a click expert to mitigate position bias in user feedback within a dual-tower design. Then, RankExpert utilizes an adaptive weight fusion layer to dynamically integrate the outputs of these specialized experts dynamically, aligning ranking predictions with diverse user intents. Extensive offline evaluations on two large-scale real-world datasets reveal that RankExpert significantly outperforms strong competitor models and substantial gains in key performance indicators. Moreover, we deploy RankExpert at Baidu Search and conduct comprehensive online evaluations with real-world web traffic, demonstrating substantial improvements in user satisfaction metrics over the online legacy system and underscoring the practicality of the proposed approach.

CCS Concepts

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

Keywords

Learning to Rank; Web Search; Mixture-of-Experts

ACM Reference Format:
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 *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/10.1145/3711896.3737258

1 Introduction

Modern learning-to-rank (LTR) systems have indeed evolved to leverage a wide range of signals beyond simple relevance-based matching [44, 58, 63]. As these systems increasingly depend on

*Corresponding author.

KDD '25, August 3–7, 2025, Toronto, ON, Canada

ACM ISBN 979-8-4007-1454-2/2025/08
https://doi.org/10.1145/3711896.3737258

both textual and behavioral features, extending pre-trained language models (PLMs) becomes essential for understanding queries and webpages at a semantic level, as well as generating end-to-end ranking score predictions [26, 61]. One way to extend PLMs for LTR tasks is to incorporate non-textual features, such as user attributes, into the ranking process [52], where authors propose the information fusion to occur at the beginning stage of the fine-tuning process, allowing the model to learn cross-information between textual and non-textual signals, such as click-through and dwell time. Another approach is to use PLMs to provide relevance and uncertainty signals for neural text ranking models, which can help to produce scale-calibrated scores [60]. In addition, LTR systems can also benefit from incorporating retrieval-augmented PLMs, which can retrieve relevant documents from a large-scale collection of documents to improve the ranking accuracy [6]. These developments highlight the importance of addressing multiple objectives, including relevance, quality, authority, recency, and so on, which reflect the multifaceted nature of user intent in today's search and recommendation platforms.

Existing methods have made strides by integrating various features into ranking models [26, 61]; however, they often fall short of fully capturing the complexity of real-world signals and face the following three significant challenges:

- *Diversity of Textual Features.* Textual inputs include additional dimensions such as content quality, authority, recency, and even uncertainty signals derived from PLMs [6, 36, 61]. While integrating non-textual features (e.g., user attributes) into PLMs during early fine-tuning shows promise, the inherent variability and multifaceted nature of textual characteristics pose significant challenges [2, 6, 52, 60]. This complexity demands that ranking models not only understand surface-level topic relevance but also capture deeper semantic nuances and contextual signals that define overall information quality.
- *Bias in Behavioral Signals.* Behavioral signals, such as clicks and dwell times, are essential for reflecting user preferences; however, these signals are often corrupted by position bias. Users tend to interact more with higher-ranked items irrespective of true relevance [9, 13, 22], leading to skewed training data. Despite the adoption of techniques such as inverse propensity scoring and specialized deep neural models for click prediction, accurately capturing genuine relevance from implicit feedback remains challenging [9, 22, 54]. The position bias not only distorts click metrics but also influences the overall fairness and accuracy of ranking outcomes.
- *Conflicts in Feature/Ranking Fusion.* Integrating heterogeneous signals–ranging from diverse textual features to debiased behavioral inputs–into a cohesive ranking model introduces conflicts in feature fusion. Existing approaches tackle information fusion using methods such as early fusion during fine-tuning [54] or score-based aggregation techniques [2, 4]. However, these methods can be computationally expensive and may not always reconcile the inherent differences between various feature types. Moreover, emerging techniques such as attention-based fusion [61] and copula-based dependency modeling [16] highlight the ongoing struggle to balance and harmonize these conflicting signals, ensuring that the integrated model accurately represents the multifaceted nature of both textual and behavioral inputs.

In response to these challenges, we propose RankExpert, a unified framework that holistically models heterogeneous ranking signals by integrating PLM-based semantic extraction with curated behavioral cues. To effectively capture the diverse textual features, encompassing relevance, quality, authority, and recency, RankExpert employs a lightweight pre-trained language model with hierarchical distillation to generate nuanced query-document representations. To address bias in behavioral signals, a dedicated click expert using a dual-tower design decouples genuine user feedback from inherent position bias. Moreover, to resolve conflicts in feature and ranking fusion, a mixture-of-experts architecture is deployed to disentangle and optimize multiple objectives independently, while an adaptive weight fusion layer dynamically integrates these outputs to harmonize their contributions into robust, debiased ranking predictions. We conduct extensive offline evaluations on two large-scale real-world datasets. Offline experimental results reveal that RankExpert substantially improves ranking accuracy in key performance metrics compared with strong baselines, while comprehensive ablation studies confirm the indispensable role of each module. We also implement the deployment of RankExpert at Baidu Search and carry out comprehensive online evaluations to further validate our approach, showing marked improvements in extensive business-side metrics. The contribution of this work could be categorized as follows.

- We identify the limitations in current ranking systems, specifically the gap between state-of-the-art PLM-driven text understanding and the integration of behavioral signals, which are critical for real-world search. Challenges including capturing heterogeneous textual features (spanning relevance, quality, authority, and recency) and mitigating systematic biases such as position bias in user interactions have been revisited in the context of PLMs. These challenges underscore the need for robust fusion mechanisms that can reconcile diverse data sources without compromising on accuracy or fairness.
- We propose RankExpert, a unified framework that integrates a lightweight PLM enhanced with hierarchical distillation, enabling efficient and effective query-document representation beyond conventional relevance. The framework features a comprehensive MoE architecture designed to disentangle and optimize multiple objectives independently, ensuring that diverse textual/behavioral signals are treated with dedicated expertise. To specifically address bias in behavioral signals, a dual-tower debiased click expert is incorporated to decouple genuine user feedback from inherent position bias. An adaptive weight fusion layer dynamically integrates outputs from all expert heads, harmonizing and balancing the contributions of textual and behavioral cues into a coherent ranking prediction.
- We conduct rigorous offline evaluations on two large-scale, real-world datasets derived from Baidu Search, demonstrating substantial improvements in key performance metrics compared to advanced competitor models. We deploy RankExpert at Baidu Search and carry out extensive online evaluations compared with the online legacy system, where RankExpert demonstrates

notable advantages in business-side metrics while preserving real-time operational efficiency and minimal latency overhead.

2 Methodology

In this section, we first describe the task formulation, and then present the overall design and technical details of RankExpert.

2.1 Task Formulation

Optimization Objective of Ranking Models. The task of ranking aims to assess the relative order among a set of 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 all indexed documents, d_i is the i^{th} document retrieved for q, and $\mathbb{Q}$ is the set of all given queries. Each document d_i is assigned a label $y_i \in \mathcal{Y}$, indicating the relevance or satisfaction degree between d_i and q, where $\mathcal{Y}$ is the set of all labels. Given the set of query-document pairs with labels, we integrate all query-document pairs with corresponding labels to perform a set of triples $\mathcal{T} = \{(q, d_i, y_i)\}_{i=1}^{|\mathcal{D}|}$. Therefore, the ranking model is required to learn a scoring function $f : \mathbb{Q} \times \mathbb{D} \rightarrow [0, 4]$, which can maximize the utility as $\max_f \mathbb{E}_{\{q,\mathcal{D},\mathcal{Y}\}} \vartheta(\mathcal{Y}, F(q, \mathcal{D}))$, where $F(q, \mathcal{D}) = \{f(q, d_i)\}_{i=1}^{|\mathcal{D}|}$ is the set of predicted ranking scores, and ϑ is the evaluation metrics, such as NDCG [18] and PNR. Therefore, the ranking model is trained to minimize the empirical loss with the labelled set as

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

where ℓ indicates the loss function of the ranking predictions of all retrieved documents in $\mathcal{D}$ of q against the ground truth.

Learning Objective of Click Models. Given q, its sorted list of candidate documents and the user's feedback (*i.e.*, *click logs*), we combine candidate documents, their corresponding positions, and users' clicks to construct a set of triples $(\mathcal{X}, \mathcal{P}, \mathbf{c})$, where $\mathcal{X} = \{x_i\}_{i=1}^{|\mathcal{X}|}$ represents the set of feature vectors of the sorted document, and x_i refers to the feature vector of the i^{th} document in the sorted list. $\mathcal{P} = \{p_i\}_{i=1}^{|\mathcal{P}|}$ is the set of position embeddings of $\mathcal{X}$, where p_i is the actual display position embedding of x_i in online search results. Moreover, $\mathbf{c} \in \{0, 1\}^{|\mathcal{X}_i|}$ is a binary vector, and $c_i \in \mathbf{c}$ indicates whether the i^{th} document is clicked in the sorted list. Given the sorted documents and their positions, click models are required to learn from the sorted list and user click logs, and estimate the conditional probability $\mathbb{P}(\mathbf{c} \mid \mathcal{X}, \mathcal{P})$ of clicks for all documents.

2.2 Framework Overview of RankExpert

Figure 1 sketches the proposed model RankExpert, which consists of four major layers: *PLM-based relevance extraction layer*, *multi-task expert learning layer*, *debiased posterior expert learning layer*, and *adaptive weight fusion layer*. Specifically, RankExpert (1) leverages a *PLM-based relevance extraction layer*, comprising two *representation encoders* and a *interaction encoder*, to extract the semantic relevance among queries, titles of corresponding documents, and content summaries of documents, and generate the relevance representation. Subsequently, RankExpert (2) separately incorporates the learned representation from *PLM-based relevance extraction layer* with diverse task features as the input, and then employs an MoE-based module to model user satisfaction comprehensively and produce the task-specific representation and scores. In *debiased click expert learning layer*, RankExpert (3) employs a dual-tower module to model users' click behaviors (using a *click tower*) and debias the position-aware bias (using a *position tower*) simultaneously, and generates the posterior representation and scores. More concretely, RankExpert integrates the learned representation from *PLM-based relevance extraction layer* with click-attribute signals (*i.e.*, *click* and *dwell time*) to construct the input for *MoE-like tower*, and models users' click and dwell time. Meanwhile, given the sorted list, corresponding positions, and click logs, RankExpert uses *position tower* to mitigate the position bias. Finally, given learned representations and predictions from the above layers, RankExpert (4) adopts an *adaptive weight fusion layer* to adaptively fuse the above layers' outputs and employ a set of high-quality satisfaction-annotated samples to yield final rankings.

2.3 PLM-based Relevance Extraction Layer

Given query-document pair (q, d_i), RankExpert first transforms q, the title and the content summary of d_i into embeddings. Subsequently, RankExpert adopts a *representation encoder*, which consists of M-layer ERNIE, to extract the semantic relevance of q and the title of d_i and generate the semantic representation as $\boldsymbol{z}_i^{qt}$. Simultaneously, RankExpert employs another *representation encoder* comprising M-layer of ERNIE to produce the semantic representation of the content summary of d_i, and generate the learned representation as z_i^s. Given the learned representation $\boldsymbol{z}_i^{qt}$ and z_i^s, RankExpert then combines them and leverages a *interaction encoder*, which contains N-layer ERNIE, to learn the interaction representation $\boldsymbol{z}_i^u$ and predicts the ranking score with a fully connected (FC) layer. In summary, RankExpert optimizes *PLM-based relevance extraction layer* using relevance-annotated training samples by minimizing a hybrid loss function integrating 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 $\mu\left(f\left(q, d_i\right), y_i\right) = \max\left\{0, \left[f\left(q, d_i\right) - \left(\frac{y_i}{5} + 0.1\right)\right]^2 - \delta\right\}$ denotes the pointwise loss. ϵ denotes the manual margin enforced between positive and negative pairs. Moreover, λ and δ represent two hyper-parameters.

2.4 Multi-Task Expert Learning Layer

Building upon the learned representation $\boldsymbol{z}_i^u$ generated by *PLM-based relevance extraction layer*, RankExpert employs an MoE-like module with four task-specific experts, each integrating $\boldsymbol{z}_i^u$ with distinct ranking features (*i.e.*, *relevance*, *quality*, *authority* and *recency*), to holistically model user satisfaction through adaptive disentanglement of heterogeneous search objectives.

Concretely, for the $k^{th}, k \in [1, 4]$ task, RankExpert maps task-specific numerical features of d_i to embedding $\boldsymbol{x}_i^k$. Next, RankExpert concatenates the first L-dimension segment of $\boldsymbol{z}_i^u$ with $\boldsymbol{x}_i^k$ to construct the input $\boldsymbol{h}_i^k$ for the k^{th} expert. Next, RankExpert implements a hybrid architecture for each expert inspired by [14], integrating

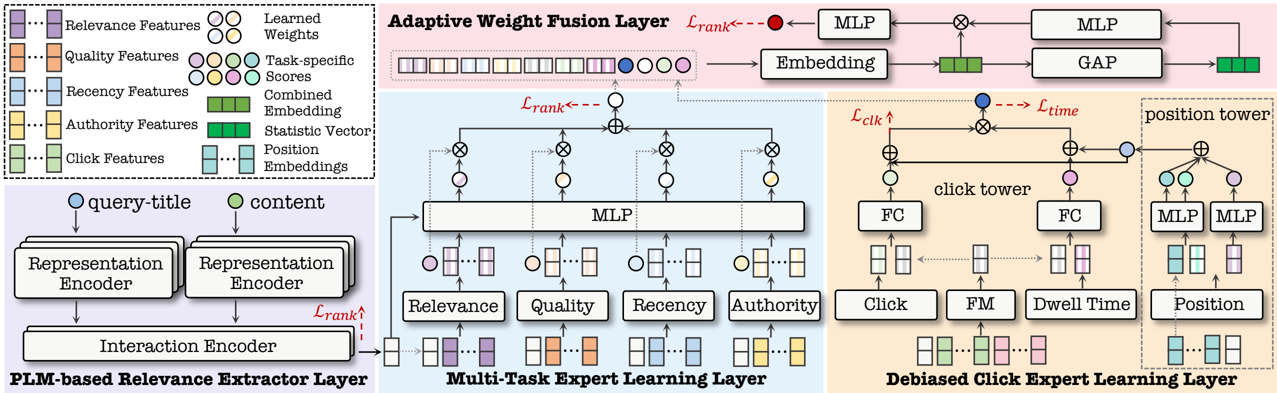

Figure 1: The overall framework of RankExpert consisting of four components: (1) PLM-based Relevance Extraction Layer, (2) Multi-Task Expert Learning Layer, (3) Debiased Click Expert Learning Layer, and (4) Adaptive Weight Fusion Layer.

two components: *factorization machine (FM) module* and *deep module*. More concretely, RankExpert employs *FM module* to compute pairwise feature relations and generate the learned representation $\boldsymbol{h}_i'^k$. In parallel, RankExpert uses *deep module* consisting of a C-layer MLP to model cross-domain dependencies and produce the representation $\overline{\boldsymbol{h}}_i^k$. Subsequently, RankExpert aggregates $\boldsymbol{h}_i'^k$ and $\overline{\boldsymbol{h}}_i^k$ into a fused representation $\widehat{\boldsymbol{h}}_i^k$, and computes the task-specific satisfaction score g_i^k via a FC layer. In particular, RankExpert optimizes the k^{th} expert using task-specific annotated training samples by minimizing the hybrid loss function integrating pairwise and pointwise objective terms as Eq (2). In this way, RankExpert separately trains four experts and obtains diverse task-specific representations and scores for d_i. Eventually, RankExpert aggregates the outputs from four experts using an MLP, which estimates task-specific adaptive weights and predicts the aggregation score as a_i'.

2.5 Debiased Click Expert Learning Layer

Given the click-attribute features (*i.e.*, *click* and *dwell time*), the sorted list, corresponding position vectors, and learned representation z_i^u generated by *PLM-based relevance extraction layer*, RankExpert leverages a dual-tower module (containing a *click tower* and a *position tower*) to model users' click behaviors and debias the position-aware bias, simultaneously.

Specifically, RankExpert first maps the click-attribute features to embeddings and concatenates them with the truncated first L-dimension of $\boldsymbol{z}_i^u$ to form the input $\boldsymbol{x}_i$ for *click tower*. Then, RankExpert uses an *FM* module to learn pairwise feature relations and generate the representation $\boldsymbol{x}_i'$. Meanwhile, inspired by [39], RankExpert implements a *click* expert and a *dwell time* expert to model users' click behaviors, simultaneously. Given the input $\boldsymbol{x}_i$, RankExpert generates the learned *click* and *dwell time* representation as $\overline{\boldsymbol{x}}_i^1$ and $\overline{\boldsymbol{x}}_i^2$ using two D-layer MLPs, respectively. Then, RankExpert computes the gate weighs as w_i^1 and w_i^2, and recombines the output of two experts as $\widehat{\boldsymbol{x}}_i^1$ and $\widehat{\boldsymbol{x}}_i^2$, where $\widehat{\boldsymbol{x}}_i^1 = w_i^1\overline{\boldsymbol{x}}_i^1 + w_i^2\overline{\boldsymbol{x}}_i^2$. Next, RankExpert separately combines $\widehat{\boldsymbol{x}}_i^1$ and $\widehat{\boldsymbol{x}}_i^2$ with $\boldsymbol{x}_i'$, and computes two distinct predictions as c_i^{clk} and c_i^{time} using two FC layers.

Meanwhile, RankExpert implements a *position tower*, which takes a cross-positional attention method [69], to mitigate the position-aware bias. More specifically, given the relevance representation x_i and position feature p_i of the i^{th} document in the ranked list, RankExpert first computes a cross-positional attention matrix to model the correlation between positions. Then, RankExpert uses the cross-position attention matrix to perform a weighted summation and calculates the cross-attention relevance representation $\tilde{x}_i$ and the cross-position feature $\tilde{p}_i$ to model the influence of surrounding documents. Given p_i and $\tilde{p}_i$, RankExpert employs an E-layer MLP to estimate the position scores r_i and $\tilde{r}_i$, respectively. In parallel, given the attended relevance feature $\tilde{x}_i$, RankExpert employs an E-layer MLP to estimate the relevance score $\tilde{g}_i$. Finally, RankExpert integrates r_i, $\tilde{r}_i$ and $\tilde{g}_i$ to compute the final position score as $r_i' = r_i + \tilde{r}_i + \tilde{g}_i$. Given the final position score r_i' from *position tower*, c_i^{clk} and c_i^{time} from *click tower*, RankExpert computes the final click score p_i^{clk} with the sum of c_i^{clk} and r_i', and the final dwell time score p_i^{time} with the sum of c_i^{time} and r_i', respectively.

In this way, RankExpert models the click by minimizing the following loss function as

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

where ζ_i is the label of the click task. Furthermore, RankExpert models the dwell time by minimizing the loss function as

$$\mathcal{L}_{time} = -\sum_{i=1}^{|\mathcal{D}|} \sigma_i \log(p_i^{clk} p_i^{time}) + (1-\sigma_i)\log\left(1-p_i^{clk} p_i^{time}\right), \tag{4}$$

where σ_i is the label of the dwell time task with $\sigma_i = 1$ representing the document has been read for more than a threshold and $\sigma_i = 0$, otherwise. Eventually, in the training phase, RankExpert optimizes *debiased click experts* by minimizing the sum of $\mathcal{L}_{clk}$ and $\mathcal{L}_{time}$.

2.6 Adaptive Weight Fusion Layer

Given the output from *multi-task expert learning layer* and *debiased click expert learning layer*, RankExpert adopts an *adaptive weight* *fusion layer* to adaptively fuse the outputs from the above layers and estimate the final satisfaction score.

Concretely, RankExpert maps the output from *multi-task expert learning layer* (*i.e.*, task-specific representations and the aggregation score g'_i) and the output from *debiased click expert learning layer* (*i.e.*, the learned representations from *click tower* and predicted results p_i^{clk}, p_i^{time} and $p_i^{clk} p_i^{time}$) to the original embedding $\boldsymbol{u}_i = [\boldsymbol{u}_i^1, \cdots, \boldsymbol{u}_i^{|\boldsymbol{u}_i|},]$. Then, RankExpert squeezes $\boldsymbol{v}_i$ into a statistic vector $\boldsymbol{e}_i = [e_i^1, \cdots, e_i^{|\boldsymbol{e}_i|}]$ using the global average pooling (GAP) function as $\boldsymbol{e}_i = \frac{1}{|\boldsymbol{u}_i|} \sum_{j=1}^{|\boldsymbol{u}_i|} v_j^i$. Subsequently, RankExpert adopts an H-layer MLP to generate attention weights for every item in $\boldsymbol{e}_i$. Next, RankExpert computes the re-weighted embedding $\boldsymbol{u}'_i$ by element-wise multiplication of the attention weights and $\boldsymbol{u}_i$. Finally, RankExpert leverages a G-layer MLP to predict the final satisfaction score $\widehat{a}_i$. In particular, RankExpert orchestrates an end-to-end optimization of the combination of four layers to achieve satisfaction-driven ranking via minimizing the mixture of pairwise and pointwise loss functions as Eq (2). Crucially, the parameters of aforementioned three layers remain frozen, while only the weights of *adaptive weight fusion layer* are updated throughout this phase.

2.7 Complexity Analysis

In this section, we present the complexity analysis and evaluate the computational complexity of each layer of RankExpert. For *PLM-based relevance extraction layer*, the computational complexity of *representation encoder* is $O\left(Mh\left(N_q + N_t\right)^2 + MhN_s{}^2\right)$, where M and h are the number of layers and hidden dimension size of ERNIE, and N_q, N_t and N_s are the length of the query, the title and the content summary of documents. Then, RankExpert employs a *interaction encoder* to generate the interaction representation. The complexity of *interaction encoder* is $O\left(Nh\left(N_q + N_t + N_s\right)^2\right)$, where N is the layer number of ERNIE. Thus, the complexity of *relevance extraction layer* is $O\left(Mh\left(N_q + N_t\right)^2 + MhN_s{}^2 + Nh\left(N_q + N_t + N_s\right)^2\right)$. For *multi-task expert learning layer*, the complexity of each expert module is $O\left(N_{exp} + Ck_c N_{exp}\right)$, where N_{exp} is the length of the input feature, and C and k_c are the number of layers and hidden dimension size of the MLP. Hence, the complexity of *multi-task expert learning layer* is $O\left(N_{exp} + Ck_c N_{exp}\right)$. For *debiased click expert learning layer*, the complexity of *click tower* is $O\left(N_{clk} + Dk_{clk} N_{clk}\right)$, where N_{clk} is the length of the input, and D and k_{clk} are the number of layers and hidden dimension size of the MLP. The complexity of *position tower* is $O\left(Ek_{pos} N_{pos}\right)$, where N_{pos} is the length of the input, and E and k_{pos} are the number of layers and hidden dimension size of the MLP. Therefore, the complexity of *debiased click expert learning layer* is $O\left(N_{clk} + Dk_{clk} N_{clk} + Ek_{pos} N_{pos}\right)$. Moreover, the complexity of *adaptive weight fusion layer* is $O\left(Hk_h N_h + Gk_g N_g\right)$, where H and G are layers of two MLPs, and k_h and k_g are the hidden dimension sizes. N_h and N_g are the lengths of the input of the two MLPs.

3 Deployment of RankExpert

In this section, we present the deployment details of RankExpert at Baidu Search: (1) *Offline Training* and (2) *Online System Workflow*.

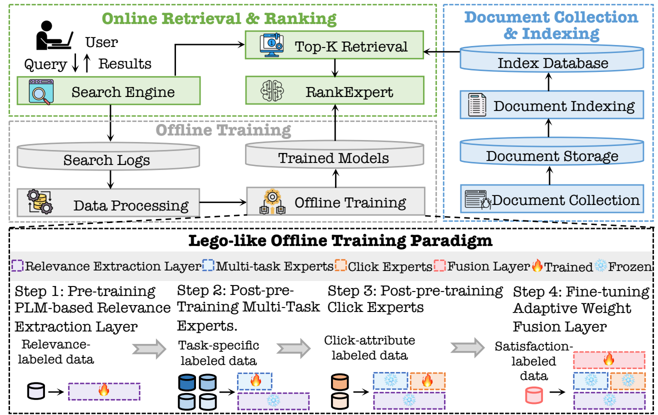

Figure 2: Deployment of RankExpert at Baidu Search.

3.1 Offline Training: A Lego-like Paradigm

As shown in Figure 2, we propose a Lego-like training paradigm that contains four steps introduced as follows.

Step 1: Pre-training PLM-based Relevance Extraction Layer. RankExpert first trains *representation encoder* to model the semantic relevance of queries and the title and the content of documents, respectively. Then, RankExpert trains *interaction encoder* to estimate ranking scores for the representation from *representation encoder*, resulting in a pre-trained relevance extraction layer.

Step 2: Post-pre-training Multi-Task Experts. RankExpert integrates *multi-task experts* with the trained *PLM-based relevance extraction layer* and trains each expert exclusively on task-specific labeled data, where RankExpert aggregates the learned representation from *PLM-based relevance extraction layer* with four task-specific features to conduct the input for the corresponding expert and separately trains each expert. Note that the parameters of *PLM-based relevance extraction layer* in this step are frozen.

Step 3: Post-pre-training Click Experts. In parallel, Given the click-attribute features and labels, the sorted list, corresponding positions, and learned representation from *PLM-based relevance extraction layer*, RankExpert trains the dual-tower module to model users' click behaviors and debias the position-aware bias, simultaneously. Similarly, the parameters of *PLM-based relevance extraction layer* in Step 3 are frozen. *In particular, owing to the unavailability of position features in the inference phase, RankExpert relies solely on the click tower of debiased click experts.*

Step 4: Fine-tuning Adaptive Weight Fusion Layer. Finally, RankExpert integrates the learned representations and predictions from *multi-task and debiased click experts* to construct the input and utilizes *adaptive weight fusion layer* to adaptively fuse the outputs and estimate the satisfaction score. In particular, RankExpert jointly trains the whole model on satisfaction-labeled data, where the parameters in the above three layers are frozen.

3.2 Online System Workflow

Referencing Figure 2, we elaborate on the RankExpert-deployed system workflow with two major components: *Document Collection & Indexing* and *Retrieval & Ranking*.

Table 1: Statistics of two offline datasets.

Dataset	BS_{pc}		BS_{mobile}	
	#queries	*#q-d pairs*	*#queries*	*#q-d pairs*
for Step 1	624,857	15,621,425	720,900	18,022,500
for Step 2	605,128	15,128,200	661,244	16,531,100
for Step 3	7,206,000	288,240,000	8,120,000	324,800,000
for Step 4	564,308	14,107,700	623,698	15,592,450

Document Collection & Indexing. To accommodate hundreds of millions of daily active users and billions of queries, Baidu Search maintains trillions of archived and indexed documents, making *document collection & indexing* indispensable. Baidu Search employs an advanced web crawler to collect documents from the internet, prioritizing newer or updated links based on real-time traffic data before downloading them. After gathering massive documents, Baidu Search uses distributed storage systems to manage them and creates efficient indices through key-value operations and in-memory computation. This setup supports high-speed retrieval.

Retrieval & Ranking. With the document database and indices in place, Baidu Search leverages an integrated retrieval model [38] that merges conventional retrieval with ERNIE-based semantic retrieval, where the conventional approach processes query text using steps like word segmentation and stop-word filtering and applies keyword matching to inverted indexes, and the ERNIE-based approach obtains a query embedding, compresses and quantizes it online, and performs retrieval. Results from both modules are combined into a candidate pool. A post-retrieval filtering stage then refines the pool via a lightweight ranker [68], which compares the candidates and retrieves statistical features from the feature database. Finally, Baidu Search executes the ranking task using RankExpert in the online ranking stage.

4 Offline Evaluation

In this section, we present the details of offline evaluation setups and then report the offline evaluation results.

4.1 Offline Datasets

We carry out the offline evaluations on two large-scale datasets collected from Baidu Search. Based on the different product lines from which the search queries originate, we divide the queries from the PC side and those from mobile devices into BS_{pc} and BS_{mobile}. The statistics of two offline datasets are illustrated in Table 1. Particularly, each query-document pair in two datasets for Step 1/Step 2/Step 4 is associated with a relevance/task-specific/satisfaction score scaled from 0 to 4 (*i.e., 0-bad, 1-fair, 2-good, 3-excellent, 4-perfect*). For Step 3, the click behavior of each pair is labeled as 1 for clicked and 0 otherwise, and the dwell time of pairs more than 40s receives a label of 1, otherwise 0. Moreover, for each stage of two datasets, samples are randomly assigned to training (80%), validation (10%), and test (10%) sets. Importantly, all offline data exclude personal user information. Furthermore, the datasets include random- and low-frequency queries. *Since this is the first study to explore satisfaction-diverse web search and incorporate textual-and-behavioral and position signals into ranking tasks, existing open-source datasets can not provide the required task-specific features and annotations.*

4.2 Evaluation Methodologies

NDCG@K (Normalized Discounted Cumulative Gain) [18] is a standard evaluation methodology for evaluating ranking models in ad-hoc search engines. *In the offline evolution, we report on the value of NDCG@4 and NDCG@10 for business and research purposes.*
PNR (Positive-Negative Ratio) is a pairwise metric in the industry community to account for the partial order among labels. PNR shows how well the predicted ranking aligns with the ground truth. For more details on these two offline evaluation methodologies, please refer to Appendix A.1.

4.3 Competitor Models

To evaluate RankExpert comprehensively, we compare it with different advanced PLM-based ranking models: **ERNIE**, **BERT** [15], **CARP** [70], **CAEHF** [71], **MFCR** [70], **KDAWD** [70], **SRM** [26] and **SRHI** [26], and advanced multi-task models: **MMoE** [39], **PLE** [51] and **M3oE** [64]. *Due to the high cost associated with deploying suboptimal models, we only compare RankExpert with the above ranking models.*

4.4 Experimental Settings

All the offline experiments are implemented on *PaddlePaddle Cloud* platform with 8 NVIDIA A100 GPUs. To perform a fair comparison, we adopt *ERNIE Lite* as the backbone network for all ERNIE-based ranking models and warm-initialize it for all ERNIE-based rankers. For *PLM-based Relevance Extraction Layer*, we assign $M = 9$ layers for two *representation encoders* and $N = 3$ layers for *interaction encoder* Moreover, we set the vocabulary size to 50,000, the hidden size to 768, the feed-forward layers with dimensions of 1024, and the batchsize to 128. For *multi-task expert*, we configure MLPs of *deep module* with $C = 3$ layers. For *debiased click expert*, we set MLPs of *click tower* and *position tower* with $D = E = 3$ layers. In *adaptive weight fusion layer*, the MLP is configured with $H = 3$ and $H = 4$ layers. All models are trained with the Adam [24] optimizer with the learning rate tuned on validation sets.

4.5 Offline Experimental Results

Overall Performance. Table 2 and 3 show the offline comparative results of RankExpert and baselines on BD_{pc} and BD_{mobile}, alongside the relative improvement over ERNIE. *Imp.* indicates the relative improvement of the model compared with the base ranker. The findings reveal the following: (1) RankExpert consistently achieves the highest performance across all three metrics on both datasets, which demonstrates its effectiveness. Specifically, RankExpert yields the largest relative improvement of 13.33%, 12.84%, and 11.96% on NDCG@4, NDCG@10 and PNR for BD_{pc}, respectively. It also achieves the highest gains of 13.01%, 11.92%, and 13.22% on these metrics for BD_{mobile}. (2) RankExpert outperforms all vanilla PLM-based ranking models and surpasses their variants, illustrating that the mixture of textual and behavioral experts strengthens PLM-based rankers to better align with satisfaction-oriented objectives. (3) RankExpert also surpasses the advanced satisfaction-oriented models SRM and SRHI. (4) Moreover, RankExpert verifies a significant lead over three multi-task frameworks, MMoE, PLE, and M3oE, on all metrics. This suggests that fusing

Table 2: Offline comparative results on NDCG@4 and NDCG@10 of RankExpert and competitor models on two offline datasets.

Model	BD_{pc}				BD_{mobile}			
	NDCG@4		NDCG@10		NDCG@4		NDCG@10	
	Value	Imp.	Value	Imp.	Value	Imp.	Value	Imp.
ERNIE	0.5602 ± 0.0024	-	0.6204 ± 0.0027	-	0.5312 ± 0.0027	-	0.5972 ± 0.0031	-
BERT	0.5646 ± 0.0035	0.79%	0.6246 ± 0.0008	0.68%	0.5355 ± 0.0032	0.80%	0.6018 ± 0.0038	0.77%
CARP	0.5734 ± 0.0032	2.36%	0.6331 ± 0.0021	2.05%	0.5387 ± 0.0018	1.39%	0.6065 ± 0.0018	1.56%
CAEHF	0.5758 ± 0.0015	2.78%	0.6355 ± 0.0034	2.43%	0.5410 ± 0.0025	1.84%	0.6084 ± 0.0021	1.88%
MFCR	0.5774 ± 0.0017	3.07%	0.6384 ± 0.0009	2.90%	0.5426 ± 0.0019	2.15%	0.6105 ± 0.0030	2.23%
KDAWD	0.5899 ± 0.0025	5.30%	0.6502 ± 0.0038	4.80%	0.5530 ± 0.0026	4.10%	0.6212 ± 0.0019	4.02%
SRM	0.5956 ± 0.0014	6.32%	0.6561 ± 0.0028	5.75%	0.5604 ± 0.0043	5.50%	0.6273 ± 0.0042	5.04%
SRHI	0.6063 ± 0.0023	8.23%	0.6668 ± 0.0017	7.48%	0.5692 ± 0.0005	7.15%	0.6355 ± 0.0046	6.41%
MMoE	0.6120 ± 0.0020	9.25%	0.6749 ± 0.0020	8.77%	0.5764 ± 0.0047	8.51%	0.6443 ± 0.0017	7.89%
PLE	0.6132 ± 0.0028	9.46%	0.6771 ± 0.0034	9.14%	0.5778 ± 0.0051	8.77%	0.6461 ± 0.0018	8.19%
M3oE	0.6176 ± 0.0007	10.25%	0.6826 ± 0.0036	10.03%	0.5821 ± 0.0020	9.58%	0.6507 ± 0.0027	8.96%
RankExpert	**0.6349 ± 0.0015**	**13.33%**	**0.7011 ± 0.0025**	**13.01%**	**0.5994 ± 0.020**	**12.84%**	**0.6684 ± 0.0035**	**11.92%**

Table 3: Offline comparative results on PNR of RankExpert and competitor models on two offline datasets.

Model	BD_{pc}		BD_{mobile}	
	Value	Imp.	Value	Imp.
ERNIE	3.352 ± 0.021	-	3.450 ± 0.009	-
BERT	3.370 ± 0.014	0.54%	3.476 ± 0.017	0.75%
CARP	3.483 ± 0.020	3.91%	3.608 ± 0.026	4.58%
CAEHF	3.498 ± 0.008	4.36%	3.630 ± 0.013	5.22%
MFCR	3.508 ± 0.015	4.65%	3.643 ± 0.024	5.59%
KDAWD	3.523 ± 0.016	5.10%	3.658 ± 0.016	6.03%
SRM	3.558 ± 0.024	6.15%	3.712 ± 0.011	7.60%
SRHI	3.601 ± 0.016	7.43%	3.753 ± 0.015	8.78%
MMoE	3.612 ± 0.014	7.76%	3.762 ± 0.027	9.04%
PLE	3.629 ± 0.005	8.26%	3.776 ± 0.010	9.45%
M3oE	3.648 ± 0.016	8.83%	3.803 ± 0.025	10.23%
RankExpert	**3.753 ± 0.019**	**11.96%**	**3.906 ± 0.007**	**13.22%**

both textual and behavioral signals is more effective than relying on textual features alone, thereby enhancing ranking accuracy.

Ablation Studies of Model Structures. To further validate the impact of four components in RankExpert, we perform comprehensive ablation experiments. We consider the following variants: RankExpert w/o *relevance extraction layer* substitutes the relevance extractor with a 12-layer ERNIE to grasp textual semantics. RankExpert w/o *multi-task experts* processes textual and multi-task inputs with *relevance extraction layer*, then feeds the representations directly to *adaptive weight fusion layer*. RankExpert w/o *debiased click experts* employs two feed-forward layers instead of learning posterior signals. RankExpert w/o *adaptive weight fusion layer* uses an MLP to integrate outputs from *multi-task experts* and *debiased click experts* before generating the final ranking score. Table 4 reports the results of removing these modules on BD_{mobile}. We find that all four components benefit the proposed model, respectively. In particular, removing *debiased click experts* causes the sharpest drop in both metrics (-11.26% and -10.17%). Moreover, we conduct further ablation studies focusing on the experts within RankExpert.

Ablation Studies of Specific Experts. We extend our ablation studies to analyze the influence of specific experts in RankExpert. Table 5 reports the ablation findings on BD_{mobile}. All experts in the proposed model significantly boost ranking performance. In particular, the relevance, quality, authority, recency, click, dwell time, and position expert separately bring increases of 0.128, 0.248, 0.250, 0.243, 0.353, 0.389, and 0.430 on PNR. Notably, excluding the position expert causes the largest performance decline, suggesting that debiasing the position-aware bias benefits satisfaction-oriented ranking. In addition, the dwell time expert displays the second-strongest effect, which reveals that dwell time can directly affect the satisfaction degree of users.

Table 4: Ablation study results of four key components in RankExpert on BD_{mobile}.

Model	PNR		NDCG@10	
	Value	Imp.	Value	Imp.
RankExpert	3.906	-	0.6684	-
- w/o *relevance extraction layer*	3.714	-4.91%	0.6415	-4.02%
- w/o *multi-task experts*	3.583	-8.27%	0.6189	-7.41%
- w/o *debiased click experts*	**3.466**	**-11.26%**	**0.6004**	**-10.17%**
- w/o *adaptive weight fusion*	3.705	-5.15%	0.6413	-4.06%

Table 5: Ablation study results of experts in *multi-task experts* and *debiased click experts* on BD_{mobile}.

Model	PNR		NDCG@10	
	Value	Imp.	Value	Imp.
RankExpert	3.906	-	0.6684	-
- w/o *relevance expert*	3.778	-3.28%	0.6487	-2.95%
- w/o *quality expert*	3.658	-6.34%	0.6290	-5.89%
- w/o *authority expert*	3.656	-6.41%	0.6307	-5.64%
- w/o *recency expert*	3.663	-6.21%	0.6320	-5.45%
- w/o *click expert*	3.553	-9.04%	0.6146	-8.05%
- w/o *dwell time expert*	3.517	-9.95%	0.6084	-8.97%
- w/o *position expert*	**3.476**	**-11.02%**	**0.6020**	**-9.93%**

5 Online Evaluation

We deploy RankExpert at Baidu Search and conduct extensive online evaluations compared with the online *legacy system*.

5.1 Interleaved Comparison

Drawing on offline experimental findings, we conducted balanced interleaving experiments to compare RankExpert and SRHI with

Table 6: Improvements of rankers compared with legacy system under interleaved comparison and manual evaluation.

Model	Random queries			Long-tail queries		
	Δ_{AB}	ΔGSB	$\Delta_{0/1}$	Δ_{AB}	ΔGSB	$\Delta_{0/1}$
legacy model	-	-	-	-	-	-
SRHI	0.37%	3.51%	-0.56%	0.59%	6.77%	-2.14%
RankExpert	**0.72%**	**6.83%**	**-3.35%**	**0.75%**	**10.18%**	**-5.87%**

All the improvements of RankExpert are statistically significant (t-test with $p < 0.05$) over *legacy system*.

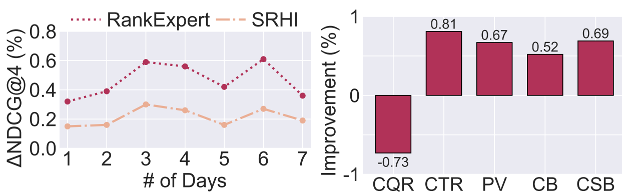

Figure 3: Performance improvement on NDCG@4 (left) and business metrics (right) of the models under online A/B Test.

legacy system in online evaluations. *Interleaving* [8] (Δ_{AB}) is a standard method in commercial search engines for assessing user preferences and summarizing experimental data. It merges the outputs of two systems and presents them to users, whose clicks are then credited to the system responsible for each clicked item. Moreover, we consider queries with different search frequencies, such as *random* and *long-tail queries*, where long-tail queries refer to those whose search frequencies are lower than 10 per week. Table 6 presents the interleaved comparison of RankExpert and SRHI against *legacy system*. RankExpert demonstrates a positive Δ_{AB}, implying more accurate ranking and increased click rates. Meanwhile, SRHI also shows benefits, indicating that textual inputs can boost satisfaction-oriented rankers. Finally, results from both random and long-tail queries confirm that RankExpert provides superior performance.

5.2 Manual Evaluations

We thoroughly investigate the impact of RankExpert through manual evaluations based on a side-by-side comparison. First, we collect a large set of queries and their top-ranked documents from the new ranking system and *legacy system* during final ranking. Then, we hire professional annotators to determine whether the new system outperforms *legacy system*. The manual evaluation results are assessed with two metrics: *Good vs. Same vs. Bad* (GSB) [65] and *Dissatisfied Ratio* ($\Delta_{0/1}$). More introduction of $\Delta_{0/1}$ can be found in Appendix B. We compare the new system deployed with trained RankExpert and SRHI, against *legacy system* and analyze different search frequencies. As presented in Table 6, RankExpert significantly enhances the online search engine, reflecting higher user satisfaction. Specifically, RankExpert attains a 6.83% improvement for random queries and a 10.18% improvement for long-tail queries on GSB. Furthermore, RankExpert gets lower $\Delta_{0/1}$ values with a -3.35% and -5.87% improvement for random and long-tail queries, indicating its ability to detect and remove dissatisfied documents from the top-4 positions.

5.3 Online A/B Test and Latency Issues

Online A/B Test. We perform a seven-day online A/B test on Baidu Search to compare the newly implemented RankExpert ranking system against *legacy system*. The A/B Test involves 0.5% of actual web traffic. From the business perspective, we evaluate NDCG@4 and determine the difference between RankExpert and *legacy system*. Figure 3 (left) shows the improvement of RankExpert and SRHI compared with *legacy model* on ΔNDCG@4. First, RankExpert could boost the performance compared with online *legacy system* on all days, which proves that RankExpert is practical for improving the performance of Baidu search engine. Moreover, RankExpert performs stably on all days, which verifies the soundness and effectiveness of our proposed model.

Impact of Business Metrics. In the online A/B Test, we concentrate on the business metrics that are directly related to user satisfaction and report the average result in Figure 3 (right). Compared to *legacy system*, RankExpert achieves -0.73% change query rates (CQR), +0.81% in click-through rates (CTR), +0.67% in page views (PV), +0.52% in click behaviors (CB), and +0.69% in click-and-stay behaviors (CSB). More results and analysis of the impact of RankExpert on business metrics can be found in Appendix C.

Latency Issues. We analyze the search log system and report the online latency of RankExpert and *legacy system*. With an online inference batch size of 10, the latencies for RankExpert and *legacy system* are approximately **11ms** and **10ms**, respectively. Although RankExpert involves a time-consuming offline training process, it introduces increased time complexity during online inference compared to *legacy system* (a 12-layer ERNIE with an MLP) due to its additional deep architecture (*multi-task experts and click experts*). Despite a modest increase in online inference time, this overhead is acceptable given the substantial gains in user satisfaction. Moreover, RankExpert operates within Baidu Search's re-ranking module, refining only the final dozens of candidate documents, thus ensuring minimal impact on overall efficiency and user experience.

6 Related Works

6.1 Textual & Behavior Signal Enhanced LTR

Conventional learning-to-rank (LTR) methods apply machine learning techniques tailored for ranking tasks and are generally classified into three categories: *pointwise* [28], *pairwise* [20] and *listwise* [46]. In recent years, PLMs have emerged as prominent tools in the LTR community [29, 35]. Concretely, Transformer-based rankers [12, 45] and BERT-based rankers [15, 23] are employed to perform ranking tasks with multiple losses. ERNIE-based rankers [30–32, 34] are also used to enhance ranking performance for industrial search engines.

Incorporating textual and behavior signals can enhance the capability of ranking models to fulfill users' various information needs. Existing ranking methods aim to improve accuracy by integrating textual features, such as *relevance*, *quality*, *authority* and *recency* [41, 59]. Specifically, textual signals pertaining to document quality have a substantial impact on ranking accuracy [10, 49, 56].

Incorporating knowledge of authoritative sources or experts enhances ranking precision by factoring in authority [37, 43]. Regarding *recency,* signals like timestamps, link times, and page classifications are included to improve rankings [11]. Beyond textual signals, behavioral signals such as users' clicks and dwell times implicitly reflect their satisfaction with the search results. Users' click behavior serves as implicit feedback on their satisfaction levels [3, 7, 33]. Moreover, the dwell time indicates their satisfaction with the provided information [40]. Previous research employs traditional statistical models [25, 48] and deep networks [55, 66] to incorporate or predict dwell time in ranking tasks. Recently, [26] strives to incorporate textual and behavior (click) signals to enhance satisfaction-oriented ranking. Moreover, incorporating position signals and addressing position bias, which assumes that documents placed higher on a page are more likely to attract clicks, are widely studied in web search [1, 21, 53]. By modeling position signals and mitigating the impact of position bias, existing ranking models can more reliably capture relevance or satisfaction [19, 69].

6.2 Mixture-of-Experts

Rooted in the divide-and-conquer strategy, the Mixture-of-Experts (MoE) framework is a method for constructing ensemble models [5] and has been widely applied in various fields [62, 67]. In web search and recommendation systems, [42] propose an MoE approach for query reformulation, utilizing multiple agents to generate and aggregate reformulated queries. [39] employs a multi-gate MoE (MMoE) to explicitly capture task relationships in multi-task learning by dynamically allocating shared and task-specific parameters through multiple gating networks. [51] presents a progressive layered extraction model for personalized recommendations, which separates shared and task-specific components and uses a progressive routing mechanism to incrementally refine semantic knowledge. To capture both long- and short-term user preferences in sequential recommendation scenarios, [64] introduces an adaptive multi-domain multi-task MoE framework (M3oE) to improve multi-domain multi-task recommendations. In the context of e-commerce platforms with data spanning multiple domains, MoE frameworks are employed to meet domain-specific requirements [27, 50, 57, 72]. Regarding ranking tasks, [27] proposes a hybrid MoE model to tackle the challenge of LTR across multiple scenarios, where data from different domains exhibit diverse characteristics. Additionally, [47] designs an MoE re-ranker that ranks sets of summary candidates, learning to select superior options and consistently enhancing the base model's performance.

Discussion. This work differentiates itself from the above research in two aspects: (1) RankExpert leverages a mixture structure of textual-and-behavioral experts to disentangle heterogeneous ranking signals to separately model different dimensions of user satisfaction. Particularly, to specifically address bias in behavioral signals, RankExpert leverages a dual-tower debiased click expert is incorporated to decouple genuine user feedback from inherent position bias. (2) RankExpert designs an *adaptive weight fusion layer* to adaptively fuse the output from diverse experts instead of directly integrating heterogeneous outputs generated from multi-task and click experts to predict the final satisfaction score.

7 Conclusion

In this work, we investigate the problem of enhancing ranking models by bridging the gap between PLM-driven text understanding and the integration of behavioral signals critical for real-world search applications, focusing on capturing heterogeneous textual features and mitigating position bias in user interactions. We propose RankExpert, a unified framework that first integrates a PLM enhanced with hierarchical distillation, enabling efficient and effective query-document representation. RankExpert then features an MoE architecture to disentangle and optimize multiple objectives independently and incorporates a dual-tower debiased click expert to address bias in behavioral signals. Finally, RankExpert uses an adaptive weight fusion layer to adaptively integrate outputs from all experts, harmonizing and balancing the contributions of textual and behavioral cues into a coherent ranking prediction. We conduct extensive offline and online evaluations using real-world web traffic from Baidu Search and compare RankExpert with numbers of competitor systems. Offline and online experimental results consistently demonstrate the effectiveness of the proposed model.

Acknowledgments

This work is partly supported by National Key Research and Development Program of China Grant 2023YFF0725100, NSFC under Grant U22B2060, Guangdong-Hong Kong Technology Innovation Joint Funding Scheme Project 2024A0505040012, the Hong Kong RGC GRF Project 16213620, RIF Project R6020-19, AOE Project AoE/E-603/18, Theme-based project TRS T41-603/20R, CRF Project C2004-21G, Guangdong Province Science and Technology Plan Project 2023A0505030011, Guangzhou municipality big data intelligence key lab, 2023A03J0012, Hong Kong ITC ITF grants MHX/078/21 and PRP/004/22FX, Zhujiang scholar program 2021JC02X170, Microsoft Research Asia Collaborative Research Grant, HKUST-Webank joint research lab and 2023 HKUST Shenzhen-Hong Kong Collaborative Innovation Institute Green Sustainability Special Fund, from Shui On Xintiandi and the InnoSpace GBA, Guangdong Basic and Applied Basic Research Foundation 2025A1515010304, and Guangzhou Science and Technology Planning Project 2025A03J4491, Yunnan Key Research Program No. 202402AD080004.

References

[1] 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.

[2] Rodger Benham, Joel Mackenzie, Alistair Moffat, and J Shane Culpepper. 2019. Boosting search performance using query variations. *ACM Transactions on Information Systems (TOIS)* 37, 4 (2019), 1–25.

[3] Alexey Borisov, Ilya Markov, Maarten De Rijke, and Pavel Serdyukov. 2016. A neural click model for web search. In *Proceedings of the 25th International Conference on World Wide Web.* 531–541.

[4] Sebastian Bruch, Siyu Gai, and Amir Ingber. 2023. An analysis of fusion functions for hybrid retrieval. *ACM Transactions on Information Systems* 42, 1 (2023), 1–35.

[5] Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2024. A survey on mixture of experts. *arXiv preprint arXiv:2407.06204* (2024).

[6] Wen-Shuo Chao, Zhi Zheng, Hengshu Zhu, and Hao Liu. 2024. Make large language model a better ranker. *EMNLP* (2024).

[7] Olivier Chapelle and Ya Zhang. 2009. A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th international conference on World wide web.* 1–10.

[8] Aleksandr Chuklin, Anne Schuth, Ke Zhou, and Maarten De Rijke. 2015. A comparative analysis of interleaving methods for aggregated search. *ACM Transactions on Information Systems (TOIS)* 33, 2 (2015), 1–38.

[9] Andrew Collins, Dominika Tkaczyk, Akiko Aizawa, and Joeran Beel. 2018. A study of position bias in digital library recommender systems. *arXiv preprint arXiv:1802.06565* (2018).

[10] Devanshu Dhyani, Wee Keong Ng, and Sourav S Bhowmick. 2002. A survey of web metrics. *ACM Computing Surveys (CSUR)* 34, 4 (2002), 469–503.

[11] Anlei Dong, Yi Chang, Zhaohui Zheng, Gilad Mishne, Jing Bai, Ruiqiang Zhang, Karolina Buchner, Ciya Liao, and Fernando Diaz. 2010. Towards recency ranking in web search. In *Proceedings of the third ACM international conference on Web search and data mining*. 11–20.

[12] Luyu Gao, Zhuyun Dai, and Jamie Callan. 2020. Modularized transfomer-based ranking framework. *arXiv preprint arXiv:2004.13313* (2020).

[13] Gizem Gezici, Aldo Lipani, Yucel Saygin, and Emine Yilmaz. 2021. Evaluation metrics for measuring bias in search engine results. *Information Retrieval Journal* 24 (2021), 85–113.

[14] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. 1725–1731.

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

[16] Jorge Hermosillo-Valadez, Eliseo Morales-González, Francis C Fernández-Reyes, Manuel Montes-y Gómez, Jorge Fuentes-Pacheco, and Juan M Rendón-Mancha. 2023. Exploiting hierarchical dependence structures for unsupervised rank fusion in information retrieval. *Journal of Intelligent Information Systems* 60, 3 (2023), 853–876.

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

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

[19] Jiarui Jin, Yuchen Fang, Weinan Zhang, Kan Ren, Guorui Zhou, Jian Xu, Yong Yu, Jun Wang, Xiaoqiang Zhu, and Kun Gai. 2020. A deep recurrent survival model for unbiased ranking. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 29–38.

[20] Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international Conference on Knowledge Discovery & Data Mining*. 133–142.

[21] 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.

[22] Changsung Kang, Xiaotong Lin, Xuanhui Wang, Yi Chang, and Belle Tseng. 2011. Modeling perceived relevance for tail queries without click-through data. *arXiv preprint arXiv:1110.1112* (2011).

[23] Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 39–48.

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

[25] Kuang-chih Lee, Burkay Orten, Ali Dasdan, and Wentong Li. 2012. Estimating conversion rate in display advertising from past erformance data. In *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*. 768–776.

[26] 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).

[27] Pengcheng Li, Runze Li, Qing Da, An-Xiang Zeng, and Lijun Zhang. 2020. Improving multi-scenario learning to rank in e-commerce by exploiting task relationships in the label space. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2605–2612.

[28] Ping Li, Qiang Wu, and Christopher Burges. 2008. McRank: Learning to Rank Using Multiple Classification and Gradient Boosting. In *Advances in Neural Information Processing Systems*. 65–72.

[29] 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.

[30] 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

[31] Yuchen Li, Haoyi Xiong, Linghe Kong, Qingzhong Wang, Shuaiqiang Wang, Guihai Chen, and Dawei Yin. 2023. S^2phere: 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. https://doi.org/10.1145/3580305.3599935

[32] 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.

[33] 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

[34] 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

[35] 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

[36] 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

[37] Yuting Liu, Bin Gao, Tie-Yan Liu, Ying Zhang, Zhiming Ma, Shuyuan He, and Hang Li. 2008. BrowseRank: letting web users vote for page importance. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. 451–458.

[38] Yiding Liu, Weixue Lu, Suqi Cheng, Daiting Shi, Shuaiqiang Wang, Zhicong Cheng, and Dawei Yin. 2021. Pre-trained language model for web-scale retrieval in baidu search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 3365–3375.

[39] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1930–1939.

[40] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. 2018. Entire Space Multi-Task Model: An Effective Approach for Estimating Post-Click Conversion Rate. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018*. 1137–1140.

[41] Jiaxin Mao, Yiqun Liu, Ke Zhou, Jian-Yun Nie, Jingtao Song, Min Zhang, Shaoping Ma, Jiashen Sun, and Hengliang Luo. 2016. When does relevance mean usefulness and user satisfaction in web search?. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 463–472.

[42] Rodrigo Frassetto Nogueira, Kyunghyun Cho, and CIFAR Global Scholar. 2018. New York University at TREC 2018 Complex Answer Retrieval Track.. In *TREC*.

[43] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank Citation Ranking: Bringing Order to the Web. In *Technical Report*. 1999–66.

[44] Georgios Peikos and Gabriella Pasi. 2024. A systematic review of multidimensional relevance estimation in information retrieval. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* (2024), e1541.

[45] Przemysław Pobrotyn, Tomasz Bartczak, Mikołaj Synowiec, Radosław Białobrzeski, and Jarosław Bojar. 2020. Context-aware learning to rank with self-attention. *arXiv preprint arXiv:2005.10084* (2020).

[46] Przemysław Pobrotyn and Radosław Białobrzeski. 2021. Neuralndcg: Direct optimisation of a ranking metric via differentiable relaxation of sorting. *arXiv preprint arXiv:2102.07831* (2021).

[47] Mathieu Ravaut, Shafiq R. Joty, and Nancy F. Chen. 2022. SummaReranker: A Multi-Task Mixture-of-Experts Re-ranking Framework for Abstractive Summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, ACL*. 4504–4524.

[48] Rómer Rosales, Haibin Cheng, and Eren Manavoglu. 2012. Post-click conversion modeling and analysis for non-guaranteed delivery display advertising. In *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012*. 293–302.

[49] Aili Shen, Bahar Salehi, Jianzhong Qi, and Timothy Baldwin. 2020. A Multimodal Approach to Assessing Document Quality. *Journal of Artificial Intelligence Research* 68 (2020).

[50] Xiang-Rong Sheng, Liqin Zhao, Guorui Zhou, Xinyao Ding, Binding Dai, Qiang Luo, Siran Yang, Jingshan Lv, Chi Zhang, Hongbo Deng, et al. 2021. One model to serve all: Star topology adaptive recommender for multi-domain ctr prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4104–4113.

[51] Hongyan Tang, Junning Liu, Ming Zhao, and Xudong Gong. 2020. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *Proceedings of the 14th ACM Conference on Recommender Systems*. 269–278.

[52] Dong Wang, Kavé Salamatian, Yunqing Xia, Weiwei Deng, and Qi Zhang. 2023. BERT4CTR: An Efficient Framework to Combine Pre-trained Language Model with Non-textual Features for CTR Prediction. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 5039–5050.

[53] 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.

[54] Yibo Wang, Yanbing Xue, Bo Liu, Musen Wen, Wenting Zhao, Stephen Guo, and Philip S Yu. 2023. Click-Conversion Multi-Task Model with Position Bias Mitigation for Sponsored Search in eCommerce. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1884–1888.

[55] Hong Wen, Jing Zhang, Yuan Wang, Fuyu Lv, Wentian Bao, Quan Lin, and Keping Yang. 2020. Entire space multi-task modeling via post-click behavior decomposition for conversion rate prediction. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 2377–2386.

[56] Gideon Maillette de Buy Wenniger, Thomas van Dongen, Eleri Aedmaa, Herbert Teun Kruitbosch, Edwin A Valentijn, and Lambert Schomaker. 2020. Structure-tags improve text classification for scholarly document quality prediction. *arXiv preprint arXiv:2005.00129* (2020).

[57] Zhuojian Xiao, Yunjiang Jiang, Guoyu Tang, Lin Liu, Sulong Xu, Yun Xiao, and Weipeng Yan. 2021. Adversarial mixture of experts with category hierarchy soft constraint. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2453–2463.

[58] 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

[59] Emine Yilmaz, Manisha Verma, Nick Craswell, Filip Radlinski, and Peter Bailey. 2014. Relevance and effort: An analysis of document utility. In *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*. 91–100.

[60] Puxuan Yu, Daniel Cohen, Hemank Lamba, Joel Tetreault, and Alex Jaimes. 2024. Explain then Rank: Scale Calibration of Neural Rankers Using Natural Language Explanations from Large Language Models. *arXiv preprint arXiv:2402.12276* (2024).

[61] Shi Yu, Chenghao Fan, Chenyan Xiong, David Jin, Zhiyuan Liu, and Zhenghao Liu. 2024. Fusion-in-T5: Unifying Variant Signals for Simple and Effective Document Ranking with Attention Fusion. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. 7556–7561.

[62] Seniha Esen Yuksel, Joseph N Wilson, and Paul D Gader. 2012. Twenty years of mixture of experts. *IEEE transactions on neural networks and learning systems* 23, 8 (2012), 1177–1193.

[63] Hamed Zamani and W Bruce Croft. 2017. Relevance-based word embedding. In *Proceedings of the 40th international acm sigir conference on research and development in information retrieval*. 505–514.

[64] Zijian Zhang, Shuchang Liu, Jiaao Yu, Qingpeng Cai, Xiangyu Zhao, Chunxu Zhang, Ziru Liu, Qidong Liu, Hongwei Zhao, Lantao Hu, et al. 2024. M3oe: Multi-domain multi-task mixture-of experts recommendation framework. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 893–902.

[65] Shiqi Zhao, Haifeng Wang, Chao Li, Ting Liu, and Yi Guan. 2011. Automatically generating questions from queries for community-based question answering. In *Proceedings of 5th international joint conference on natural language processing*. 929–937.

[66] Zhong Zhao, Yanmei Fu, Hanming Liang, Li Ma, Guangyao Zhao, and Hongwei Jiang. 2021. Distillation based multi-task learning: A candidate generation model for improving reading duration. *arXiv preprint arXiv:2102.07142* (2021).

[67] Zhe Zhao, Lichan Hong, Li Wei, Jilin Chen, Aniruddh Nath, Shawn Andrews, Aditee Kumthekar, Maheswaran Sathiamoorthy, Xinyang Yi, and Ed Chi. 2019. Recommending what video to watch next: a multitask ranking system. In *Proceedings of the 13th ACM conference on recommender systems*. 43–51.

[68] Zhaohui Zheng, Keke Chen, Gordon Sun, and Hongyuan Zha. 2007. A regression framework for learning ranking functions using relative relevance judgments. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. 287–294.

[69] 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.

[70] Lixin Zou, Weixue Lu, Yiding Liu, Hengyi Cai, Xiaokai Chu, Dehong Ma, Daiting Shi, Yu Sun, Zhicong Cheng, Simiu Gu, et al. 2022. Pre-trained language model-based retrieval and ranking for web search. *ACM Transactions on the Web* 17, 1 (2022), 1–36.

[71] Lixin Zou, Shengqiang Zhang, Hengyi Cai, Dehong Ma, Suqi Cheng, Shuaiqiang Wang, Daiting Shi, Zhicong Cheng, and Dawei Yin. 2021. Pre-trained language model based ranking in Baidu search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 4014–4022.

[72] Xinyu Zou, Zhi Hu, Yiming Zhao, Xuchu Ding, Zhongyi Liu, Chenliang Li, and Aixin Sun. 2022. Automatic expert selection for multi-scenario and multi-task search. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1535–1544.

A Offline Evaluation Setups

A.1 Offline Evaluation Methodologies

To assess the performance of different ranking models comprehensively, we adopt the evaluation metrics as follows:

- **Normalized Discounted Cumulative Gain (NDCG)** [18] is a standard listwise accuracy metric, which has been widely adopted to evaluate the search relevance. Given a query q and the retrieved document set $\mathcal{D}$, the ranking model usually predicts a score for each webpage and generates a ranking list by sorting scores in descending order. The NDCG score could be computed as

$$NDCG@K = \frac{1}{Z} \sum_{i=1}^{|\mathcal{D}|} \frac{2^{y_i} - 1}{\log_2(1+i)}, \tag{5}$$

  where Z is a normalization factor that is the ideal order of Discounted Cumulative Gain (DCG) [17], and y_i is the ranking score of the i^{th} document. Moreover, the value of NDCG falls between 0 and 1, with higher values indicating superior performance. In our offline experiments, we report the NDCG result of top-4 and top-10 (i.e., NDCG@4 and NDCG@10) for business and research purposes.
- **Positive-Negative Ratio** (PNR) is a simple yet effective metric for evaluating the search relevance performance in the industry, which indicates the consistency of the ranking score with the ground truth. Given a query q and its ranked documents $\mathcal{D}$, PNR is the ratio of concordant pairs to discordant pairs as

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

  where $\mathbb{1}\{x > y\}$ is an indicator function equal to 1 if $(x > y)$, and 0 otherwise. PNR shows how well the predicted ranking aligns with the ground truth. In the offline experiments, the symbol PNR represents the average value across a set of test queries.

B Online Evaluation Methodologies: Dissatisfied Ratio

Dissatisfied Ratio ($\Delta_{0/1}$) is a widely used online metric to assess users' disappointment, which is a ratio of dissatisfied results at the top-K position of the first page, calculated as

$$\Delta_{0/1} = \frac{\sum_{i=1}^{K} \mathbb{1}\left\{y_i = 0 \ or \ 1\right\}}{K}. \tag{7}$$

Unlike the above online metrics, a lower value of $\Delta_{0/1}$ signifies the superior ranking performance of the corresponding model. Given that the top four results are critically important for user impressions, our online ranking system typically reports outcomes for $K = 4$.

C Impact of Business-side Metrics

Deploying new ranking models in large commercial search engines like Baidu Search relies on specific business metrics as critical indicators of their effectiveness in enhancing user satisfaction. In the online A/B test, we concentrated on metrics directly related to user consumption and reported the improvements of RankExpert compared with the *legacy system*. In particular, RankExpert achieved the following relative improvements over the *legacy system*:

- Satisfaction Consumption Rate (SCR): increased by 3.13%.
- Change Query Rate (CQR): decreased by 0.73%.
- Click-through Rate (CTR): increased by 0.81%.
- Average Post-click Dwell Time (APCDT): increased by 0.75%.
- Page Views (PV): increased by 0.67%.
- Click Behaviors (CB): increased by 0.52%.
- Click-and-Stay Behaviors (CSB): increased by 0.69%.

Aligned with the aforementioned metrics, the proposed model demonstrates substantial improvements over the *legacy system* in both revenue and user engagement metrics. All the reported values are statistically significant with $p < 0.05$. These findings indicate that the proposed model significantly enhances user engagement within the search engine context.