Deep Learning to Rank in Industrial Search Engines, Recommender Systems, and Online Advertising: An Overview and New Perspectives

YULONG GU, Tsinghua University, Beijing, China
LIXIN ZOU and CHENLIANG LI, Key Laboratory of Aerospace Information Security and Trusted Computing, School of Cyber Science and Engineering, Wuhan University, Wuhan, China

Search engines, Recommender systems, and Online advertising are playing fundamental roles in modern web and mobile applications. In these information systems, the most significant component is the ranking system, which selects a list of items likely to interest a user from billions of candidate items. At its core, Deep learning to Rank (DLTR) has become indispensable for building high-performance ranking models, driving significant gains in user engagement and business growth. In this article, firstly, we outline the key problems and challenges in industrial-scale ranking systems. Secondly, we provide a comprehensive review of deep learning models deployed across multiple stages of the industrial ranking pipeline, including matching, pre-ranking, fine-grained ranking, post-ranking, and relevance-ranking. Finally, we explore novel perspectives for future research, such as leveraging Large Language Models (LLMs). The papers discussed in this survey are listed in https://github.com/guyulongcs/Awesome-Deep-Learning-Papers-for-Search-Recommendation-Advertising.

CCS Concepts: • **Information systems** → **Information retrieval**; **Recommender systems**; **Computational advertising**;

Additional Key Words and Phrases: Deep Learning, Information Retrieval, Search Engines, Recommender Systems, Online Advertising, Matching, Ranking, Relevance, Click-Through Rate Prediction, Conversion Rate Prediction

ACM Reference format:
Yulong Gu, Lixin Zou, and Chenliang Li. 2026. Deep Learning to Rank in Industrial Search Engines, Recommender Systems, and Online Advertising: An Overview and New Perspectives. *ACM Trans. Inf. Syst.* 44, 4, Article 83 (April 2026), 52 pages.
https://doi.org/10.1145/3797895

1 Introduction

The proliferation of web and mobile applications (e.g., Google [124], Meta [129], YouTube [38], Amazon [109], TikTok, Douyin [21]) has profoundly transformed and enriched daily life. However,

We express our sincere gratitude for the financial support provided by the National Natural Science Foundation of China (No. U23A20305 and No. 62302345) and the Xiaomi Young Scholar Program.
Authors' Contact Information: Yulong Gu (corresponding author), Tsinghua University, Beijing, China; e-mail: guyulongcs@gmail.com; Lixin Zou, Key Laboratory of Aerospace Information Security and Trusted Computing, School of Cyber Science and Engineering, Wuhan University, Wuhan, China; e-mail: zoulixin@whu.edu.cn; Chenliang Li, Key Laboratory of Aerospace Information Security and Trusted Computing, School of Cyber Science and Engineering, Wuhan University, Wuhan, China; e-mail: cllee@whu.edu.cn.

ACM 1558-2868/2026/4-ART83
https://doi.org/10.1145/3797895

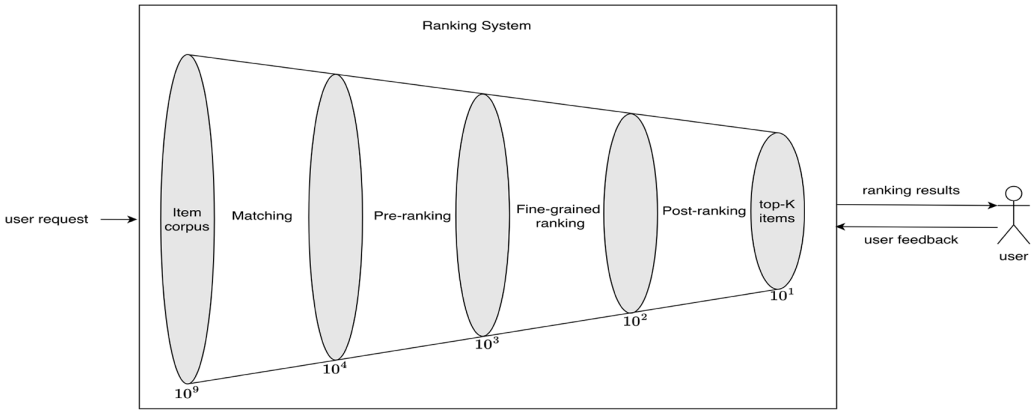

Fig. 1. The multi-stage cascading ranking system in industrial search engines, recommender systems, and online advertising.

this proliferation has exacerbated the problem of information overload, as these platforms host millions or even billions of items.

To address this problem, industrial Search engines, Recommender systems, and Online advertising employ sophisticated ranking systems to deliver a personalized list of top-K items from billions of candidates to each user within hundreds of milliseconds. Industrial ranking systems face several critical challenges: providing real-time personalized results under strict latency constraints, managing the immense computational complexity of evaluating billions of items for billions of users, and working within hardware resource limitations.

To tackle these challenges, industrial ranking systems typically adopt a Multi-stage Cascading Ranking System paradigm (Figure 1), which decomposes the ranking task into sequential stages (e.g., matching, pre-ranking, fine-grained ranking, and post-ranking stages), where each stage makes a different balance between efficiency and accuracy. Specifically, each stage progressively employs more sophisticated models to narrow down the candidate set, for instance, reducing the pool from 10^9 items in the corpus to 10^4 after matching, 10^3 after pre-ranking, 10^2 after fine-grained ranking, and finally to around 10 items after post-ranking.

Ranking models in the Multi-stage Cascading Ranking System are typically built using **Learning to Rank (LTR)** [112], which leverages machine learning to learn models based on user feedback [74, 93, 112, 124, 211]. Initially, classical approaches such as **Logistic Regression (LR)** [124] and **Gradient Boosting Decision Trees (GBDT)** [60] were widely adopted in industrial LTR before the deep learning revolution [94, 96]. Since 2012, **Deep LTR (DLTR)** [38, 60, 138, 210, 240] has achieved remarkable success in industrial ranking systems, driving substantial performance gains over traditional methods. More recently, the rise of **Large Language Models (LLMs)** (e.g., GPT [15], GPT-4 [2], ChatGPT [131], LLaMA [174]) in **Natural Language Processing (NLP)** has introduced new possibilities for ranking (e.g., GR [215], TIGER [148]).

This survey provides a comprehensive review of DLTR approaches in industrial ranking systems. First, we outline key problems and challenges in industrial ranking systems, and review the traditional methods that were prevalent prior to the deep learning era (i.e., before 2012). Second, we review deep learning-based approaches which are widely deployed in industrial Multi-stage Cascading Ranking Systems between 2012 and 2026, focusing on methods validated through online A/B testing in large-scale industrial environments. Third, we explore emerging paradigms such as LLMs-based ranking models, which have gained prominence since 2022. Finally, we discuss

open research problems (e.g., Cold-start problem [39], The Feedback loop problem [58], Long-term reward optimization problem [248]) and research directions (e.g., LLM-based ranking) that are critical for future advancements.

Differences from Previous Surveys. Although several existing surveys [34, 46, 97, 102, 106, 190, 225] provide valuable overviews of deep learning for ranking, they exhibit several critical limitations. First, they are primarily written from an academic perspective, often neglecting the practical constraints and challenges of industrial-scale systems. Many of the methods discussed are validated only on small-scale public datasets, and their efficacy in large-scale industrial environments remains unproven. Second, they lack coverage of recent advances in industrial ranking systems such as LLM-based ranking models, as many of them were published several years ago. While some recent surveys [46, 106, 190] focus exclusively on LLM-based approaches, they often omit other crucial deep learning paradigms, and the methods they discuss are frequently not validated in industrial settings.

Methods for Collecting Papers. For this survey, we used Google Scholar to identify relevant literature on Search engines, Recommender systems, and Online advertising, which were published in top-tier conferences (e.g., KDD, SIGIR, WWW, CIKM, ICDM, RecSys, ICLR, NeurIPS, ICML, AAAI, IJCAI) and journals (e.g., TOIS, TKDE) from 2000 to 2026. Our search keywords include "information retrieval," "recommender system," "online advertising," "matching," "ranking," "post ranking," "relevance," "deep learning," and "large language models." A key filter was our requirement that selected approaches must have their effectiveness validated through online A/B testing in large-scale industrial environments. This selection criterion ensures that this survey contains state-of-the-art methods proven in real-world industrial ranking systems.

In summary, the key contributions of this survey are:

- We introduce a systematic outline of the fundamental problems and challenges in industrial ranking systems.
- We provide the first comprehensive review of DLTR approaches validated and deployed in industrial settings.
- We offer an in-depth review of the emerging LLM-based ranking methods, examining their promising abilities (e.g., semantic understanding, scaling laws) and integration challenges within existing industrial architectures.
- We highlight critical open problems and future research directions essential for advancing the state of the art in industrial ranking.

Outline of This Article. As shown in Figure 2, the remainder of this article is structured as follows: Section 2 introduces preliminaries and key problems of industrial ranking systems in search, recommendation, and online advertising. Section 3 provides an overview of deep learning models within the multi-stage ranking paradigm. Sections 4–8 present detailed reviews of deep learning approaches for matching, pre-ranking, fine-grained ranking, post-ranking, and relevance-ranking, respectively. Section 9 discusses evaluation datasets and metrics. Section 10 explores future research directions and open problems. Section 11 concludes the survey.

2 Overview of Ranking Systems in Search Engines, Recommender Systems, and Online Advertising

In this section, we provide an overview of ranking systems in Search engines, Recommender systems, and Online advertising. Section 2.1 introduces the overview of Search engines, Recommender systems, and Online advertising. Section 2.2 details an overview of ranking systems in these information systems. Finally, Section 2.3 introduces DLTR, the dominant methodology for learning ranking models in industrial ranking systems.

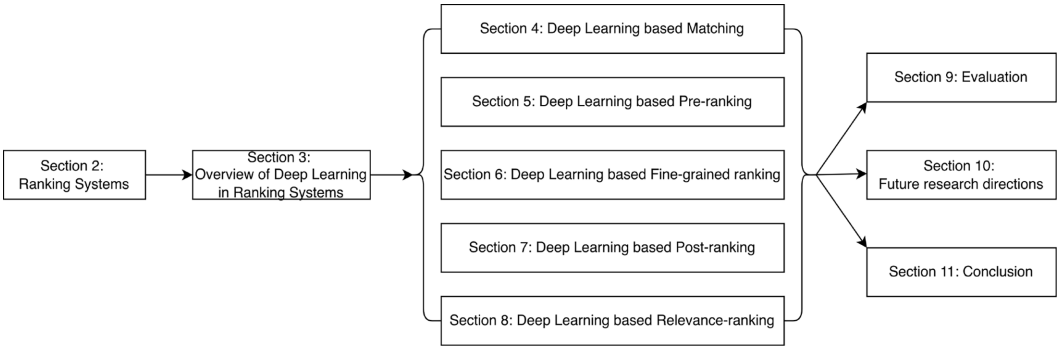

Fig. 2. Outline of this article.

2.1 Search Engines, Recommender Systems, and Online Advertising

Search engines, Recommender systems, and Online advertising, which select a list of items likely to interest a user from billions of candidate items, serve as the primary engines for user growth and business growth, playing vital roles in modern web and mobile applications. These technologies are foundational to a wide range of platforms, including short-video services (TikTok, YouTube Shorts, Douyin, Kuaishou), E-commerce (Amazon, Temu, TikTok Shop, Alibaba, Pinduoduo, JD.com), video streaming (YouTube, Netflix), image-sharing platforms (Instagram, Pinterest), social networks (Meta, WeChat, X, Reddit), local-life services (Yelp, Meituan), question-and-answer sites (Quora), and news sites (Google News). They can enhance user engagement (e.g., satisfaction and retention) and drive revenue growth via product sales and online advertising.

This section introduces the close relationships between Search engines, Recommender systems, and Online advertising in Section 2.1.1. Building on this foundation, Section 2.1.2 explains our motivation for discussing them within a unified survey. Section 2.1.3 then compares industrial and academic research in ranking, highlighting the importance of online A/B testing in industrial ranking systems. Consequently, this survey focuses specifically on approaches that have demonstrated success in large-scale industrial applications.

2.1.1 Relationships between Search Engines, Recommender Systems, and Online Advertising. The fields of Search engines, Recommender systems, and Online advertising are deeply interconnected. Search engines (e.g., Google Search, Amazon Search) can be viewed as specialized Recommender systems that employ relevance models (e.g., MASM [206], ReprBERT [207]) to select items relevant to the user's query. Online advertising is broadly categorized into search advertising and recommendation advertising. The former can be understood as a specialized application of search engine technology, while the latter applies recommender system techniques, with both systems designed to select and present advertisements. Consequently, the underlying ranking systems across these domains share the common paradigm.

2.1.2 Motivation for Discussing Ranking in Search Engines, Recommender Systems, and Online Advertising in a Unified Survey. Although Search engines, Recommender systems, and Online advertising are often treated as three domains, this survey integrates them based on four fundamental commonalities:

— A unified problem formulation. The core problem in Search engines, Recommender systems, and Online advertising is essentially the same: effectively ranking a massive corpus of items

in response to a user's request. This shared objective provides a common foundation for analyzing ranking techniques.
— A shared architectural paradigm. Industrial ranking systems across these domains typically follow the same architectural framework: The Multi-stage Cascading Ranking System. This pipeline, typically consisting of Matching, Pre-ranking, Fine-grained ranking, and Post-ranking stages, is a universal design pattern for scalable ranking across all three domains.
— Cross-domain technical synergy. The deep learning models employed in these domains are highly transferable. For instance, the Two-Tower architectures are standard for efficient matching, while the Embedding and **Multi-Layer Perceptron (MLP)** paradigm forms the backbone for fine-grained ranking models, enabling direct knowledge transfer.
— The value of a holistic view. A domain-specific perspective often leads to a fragmented understanding. By integrating these fields, this survey provides a complete and coherent picture of industrial ranking, providing researchers and practitioners with a unified technical framework that fosters cross-domain innovation.

2.1.3 Comparison of Industrial and Academic Research in Search Engines, Recommender Systems, and Online Advertising. Industrial and academic research in Search engines, Recommender systems, and Online advertising both aim to improve ranking based on machine learning. However, they differ in several key aspects:

— Objectives. Academic research is driven by conceptual breakthroughs and exploration of new directions, guiding long-term progress. In contrast, industrial research focuses on enhancing the performance and profitability of online services.
— Evaluation method. Academic research typically evaluates models on offline public datasets using metrics such as Recall@K and **Area Under the Receiver Operating Characteristic Curve (AUC)**. The public datasets are small-scale (e.g., millions of examples). While useful for controlled comparisons, these methods fail to capture the complexities of dynamic production environments. Industrial practice treats offline results as a preliminary check, with ultimate validation coming from online A/B tests that measure improvement in online metrics. In offline testing, it uses large-scale industrial datasets (e.g., billions of examples) collected from online services and uses offline metrics to validate the model's effectiveness. In online A/B testing, online metrics (e.g., **Gross Merchandise Volume (GMV)** and orders) are used to evaluate the model's final performance in online services. Due to the gap in evaluation, some methods in academic research may not be effective in industrial scenarios.
— Consideration of system limitations and hardware costs. Academic research methods often neglect system limitations and hardware costs. However, industrial research methods need to thoroughly consider these problems. For example, industrial ranking systems have strict latency constraints (e.g., a few hundred milliseconds). Industrial ranking systems often need to make a tradeoff between performance improvement, system latency, and hardware costs.

Due to the gap between academic and industrial research in ranking, some academically popular approaches have seen limited adoption in industrial ranking systems, such as **Deep Reinforcement Learning (DRL)**-based ranking.

— DRL-based ranking. DRL-based ranking methods [41, 234, 248, 250] are popular in academic research. These approaches treat ranking as a sequential decision-making problem. The goal is to learn a policy that generates a list of items based on the user's state to maximize long-term cumulative reward (e.g., total time spent on the platform). Despite its theoretical appeal, they face significant challenges in industrial settings: (1) The scales of state space and action space in these models are extremely large because there are billions of items in industrial ranking

systems. The state is usually represented as the sequence of historically interacted items, and the action space encompasses the entire item corpus. This leads to unstable training and difficulties in model convergence. (2) On-policy-based models are sample inefficient. They need a massive amount of interaction data to learn from the current policy. The exploration needed to collect this data can severely degrade the user experience and harm key online metrics during training. (3) Off-policy-based models can use historical data. However, accurately estimating the performance of a new policy using logs generated by a different production policy is notoriously difficult and often unreliable.

Therefore, to demonstrate practical utility, models in academic research must ultimately be validated through large-scale online A/B testing.

This survey focuses on approaches that have been rigorously tested and proven successful in large-scale industrial ranking systems, offering researchers in academia and industry a clearer understanding of practical challenges and proven techniques.

2.2 The Ranking System in Industrial Information Systems

In these industrial information systems (i.e., Search engines, Recommender systems, and Online advertising), the core component is the ranking system, responsible for generating a personalized list of top-K items for each user in real time from a large item corpus.

This section introduces the definition of the ranking problem in Section 2.2.1. It then introduces the evaluation metrics in Section 2.2.2 and discusses the key challenges of the industrial ranking system in Section 2.2.3. Finally, it presents the Multi-stage Cascading Ranking System paradigm in Section 2.2.4.

2.2.1 Definition of the Ranking Problem. Consider a set of m users $\mathcal{U} = \{u_1, u_2, ..., u_m\}$ and a set of n items $\mathcal{I} = \{i_1, i_2, ..., i_n\}$. Each user u_i has a list of interacted items $\mathcal{I}_{u_i}$, which the user has expressed his/her feedback on. This feedback can be explicit, such as a rating score, or implicit, inferred from users' behaviors (e.g., click, add to cart, purchase behaviors in E-commerce; stay, like, follow, collect, share, comment behaviors in content applications).

Definition 2.1. The Ranking Problem: For a user $u \in \mathcal{U}$, the ranking problem is to generate an ordered list of top-K items $[i_1, i_2, ..., i_K] (i_j \in \mathcal{I}, j \in [1, K])$ from the entire item set $\mathcal{I}$.

2.2.2 Evaluation Metrics of Industrial Ranking System. For an industrial ranking system, the evaluation metrics consist of performance metrics and efficiency metrics. The design of a ranking system typically involves a tradeoff between performance and efficiency. While more complicated ranking models often achieve better business metrics, they have the disadvantages of higher computational cost, increased latency, and reduced throughput.

Performance Metrics of Industrial Ranking System. The primary goal of a ranking system is to optimize online business performance metrics. These metrics are domain-specific: e-commerce platforms focus on Orders and GMV [21, 105]; web search engines prioritize Clicks and Query-Change Rate [115]; content platforms track metrics like user's Active Days, Stay time, and Engagement metrics [245]; and online advertising systems optimize for Ad Revenue and **Effective Cost Per Mille (eCPM)** [245]. These online metrics are detailed in Section 9.4.

Efficiency Metrics of Ranking Systems. The key metrics for evaluating computational efficiency are as follows:

— **Floating-Point Operations (FLOPs)**. In machine learning, a model's FLOPs represent the number of FLOPs required for a single inference, indicating its computational complexity.

—Latency. Latency refers to the total time taken to process a user request and return a ranking result.
—Throughput. Throughput is the number of user requests a system can process per unit of time.

2.2.3 Challenges of the Ranking System. Industrial ranking systems must overcome several critical challenges:

—Ranking quality. The system must identify a small set of highly relevant items for a user from a pool of billions, as the quality of results directly impacts user satisfaction and revenue.
—Scalability. The system must efficiently handle millions or even billions of users and items.
—Low latency. Personalized rankings must be generated under strict latency constraints, typically within a few hundred milliseconds.
—Dynamic environment. The ranking system must rapidly adapt to shifting user interests and new items.
—Hardware constraints. Deployment is constrained by the high cost of computational (e.g., GPU, CPU) and storage (e.g., memory) resources required to serve billions of users.

The combined constraints of high hardware costs and low latency make it infeasible to score billions of items per user using a single complex ranking model.

2.2.4 The Multi-Stage Cascading Ranking System. To address these challenges, industrial ranking systems typically adopt the Multi-stage Cascading Ranking System paradigm, which consists of several ranking stages: matching, pre-ranking, fine-grained ranking, post-ranking, and relevance-ranking (as shown in Figure 1).

—Matching (candidate generation). This stage retrieves a subset of N_1 (e.g., $N_1 = 10^4$) items from all the N (e.g., $N = 10^9$) items in the corpus. Industrial matching modules typically employ multiple matching methods (e.g., collaborative filtering, embedding-based) in parallel and blend their resulting candidate sets.
—Pre-ranking. This stage further narrows down the candidate set from the matching stage (e.g., from $N_1 = 10^4$ to $N_2 = 10^3$ items) using a faster, lighter-weight ranking model.
—Fine-grained ranking. This core ranking stage applies a more complex and accurate model to the pre-ranked candidates to select a shortlist (e.g., $N_3 = 10^2$ items).
—Post-ranking. This stage finalizes the top-K items (e.g., $K = 10^1$) from the ranking output, often incorporating diversity constraints or business factors.
—Relevance-ranking. The Relevance-ranking model [80, 206, 211] calculates the relevance score between a user's query and candidate items. It is critical for search engines, but is generally not required for pure recommender systems. Relevance-ranking can be utilized within the matching, pre-ranking, or fine-grained ranking stages.

The multi-stage cascading paradigm enables the efficient generation of high-quality item lists from corpora of millions or billions of items. Downstream stages (e.g., fine-grained ranking) usually employ more accurate but computationally expensive models, resulting in higher latency compared to upstream stages (e.g., matching). The total end-to-end latency for an industrial ranking system typically is less than several hundred milliseconds. The key advantage of this paradigm is the ability to balance ranking quality, hardware cost, and latency by strategically adjusting model complexity and candidate set size at each stage.

In each stage of the multi-stage cascading paradigm, ranking is performed based on ranking models and ranking strategies.

The Ranking Models. Within each stage, ranking models are learned to estimate key unilateral scores (e.g., **Click-Through Rate (CTR)**, **Conversion Rate (CVR)**, or Dwell Time [38]) for each candidate item.

The Ranking Strategies. In each ranking stage, a ranking strategy (or formula) combines these unilateral scores into a final ranking score for each item. The ranking strategies are domain-specific.

In e-commerce search and recommendation, the final ranking score s is often defined to jointly optimize for clicks (i.e., CTR), orders (i.e., CTR · CVR), and GMV (i.e., Gross Merchandise Value, CTR · CVR · price) [60, 228, 246], as shown in Equation (1) or (2):

$$ranking\_score = \alpha_0 * CTR + \alpha_1 * CTR * CVR + \alpha_2 * CTR * CVR * price, \tag{1}$$

$$ranking\_score = (1 + \alpha_0 * CTR)^{\beta_0} * (1 + \alpha_1 * CTR * CVR)^{\beta_1} * (1 + \alpha_2 * CTR * CVR * price)^{\beta_2}, \tag{2}$$

where α and β are tunable weight parameters, and price denotes the selling price of the item.

For content platforms (e.g., YouTube, TikTok, Douyin, Tencent Videos), the ranking score is typically designed to maximize user engagement, as defined in Equation (3):

$$ranking\_score = \prod_{i=1}^{I} (1 + \alpha_i * CVR_i)^{\beta_i}, \tag{3}$$

where each CVR_i corresponds to an interaction signal (e.g., completion rate, follow rate, like rate, collection rate, share rate, comment rate, or dwell time) [171], and α and β are weight parameters that control the influence of each signal.

In online advertising, the ranking score is typically the estimated eCPM (Equation (4)), which directly maximizes platform ad revenue [240]:

$$ranking\_score = eCPM = \begin{cases} CTR \cdot bid \cdot 1{,}000 & : Cost\text{-}Per\text{-}Click\ Ads, \\ CTR \cdot CVR \cdot bid \cdot 1{,}000 & : Cost\text{-}Per\text{-}Action\ Ads, \end{cases} \tag{4}$$

where *bid* is the advertiser's bid price of the ad. Revenue in the **Cost-Per-Click (CPC)** pricing model is based on clicks, while in the Cost-Per-Action pricing model, it is based on conversions.

The accuracy of core ranking models (e.g., CTR, CVR prediction) has a direct and significant impact on key online business metrics (e.g., User Retention, GMV, Advertising revenue) [240] because their ranking scores are directly used in the ranking strategy (i.e., ranking formulas).

2.3 DLTR in Ranking Systems

In Multi-stage Cascading Ranking Systems, DLTR is widely employed to learn the ranking models within each stage.

2.3.1 LTR. LTR [112] applies machine learning to learn ranking models that select top items from a candidate set. Traditional LTR methods (e.g., **Matrix Factorization (MF)** [93], LR [124], GBDT [211], and LR and GBDT [74]) have achieved great success in learning ranking models.

2.3.2 DLTR. Deep learning [96] has driven remarkable progress across AI domains, including NLP, Computer Vision, and Speech Recognition. Inspired by this success, DLTR leverages **Deep Neural Networks (DNN)** to learn ranking models, leading to substantial performance gains in industrial systems since 2012 [38, 60, 210, 240]. Most recently, methods based on LLMs have shown promising results for ranking tasks [51, 159, 215].

Model Architecture of Deep Ranking Models. Deep ranking models are built using popular deep learning components, such as Embedding, MLP, and Transformer (shown in Figures 3–6).

Labels of Deep Ranking Models. These models are trained using implicit or explicit feedback signals from user interactions. In E-commerce sites, common labels include clicks, add-to-cart

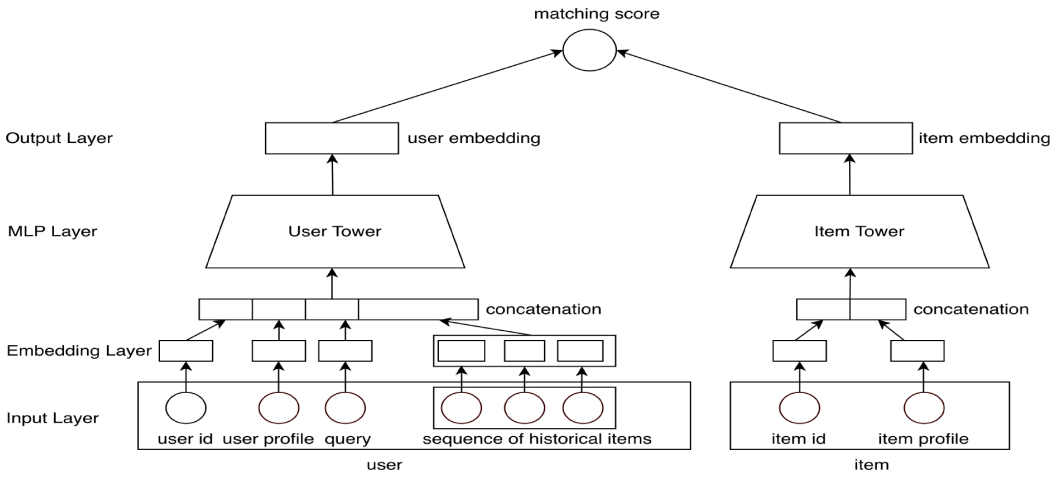

Fig. 3. The Two-Tower architecture for matching.

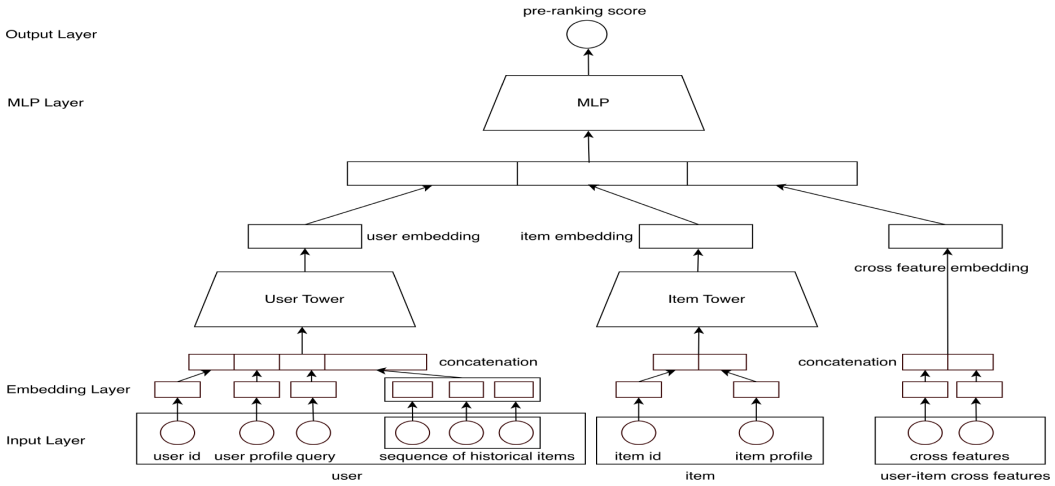

Fig. 4. The Two-Tower and MLP architecture for pre-ranking.

actions, and purchases. In short-video platforms, labels include users' dwell time, share, comment, and collect behaviors on short-videos.

Features of Deep Ranking Models. The input features for deep ranking models typically encompass user profiles [61], user behavior history, item profiles, contextual signals, queries (in search), and cross-features.

- —User profile. It consists of the user's demographic and static information, such as user ID, age, and gender.
- —Users' historical behaviors. They contain a sequence of items a user has historically interacted with in the system.
- —Item profile. It consists of the candidate (i.e., target) item's attributes, such as item ID, category, and textual description.

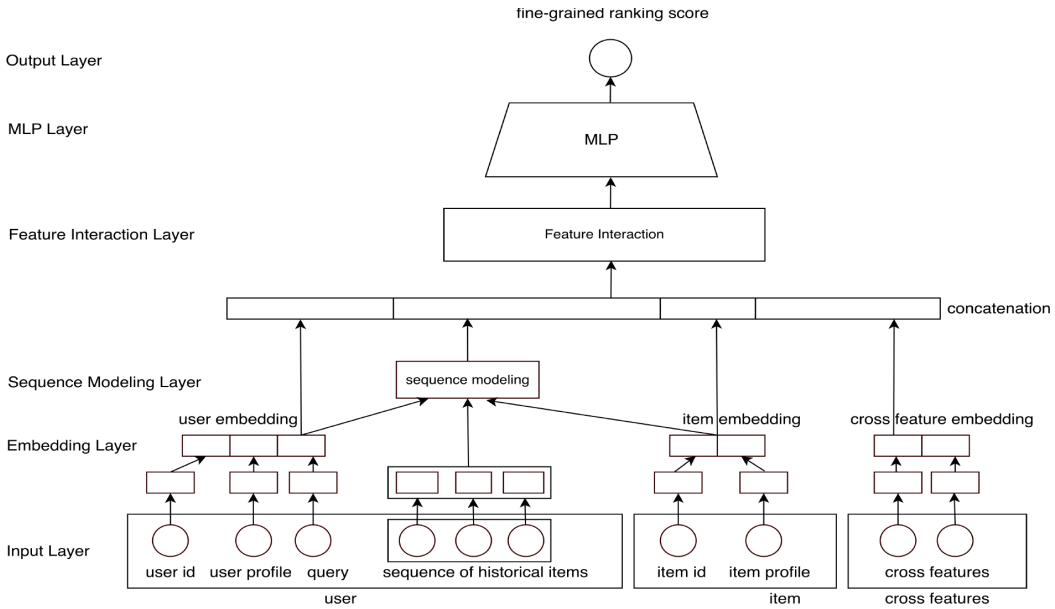

Fig. 5. The embedding and MLP architecture for fine-grained ranking.

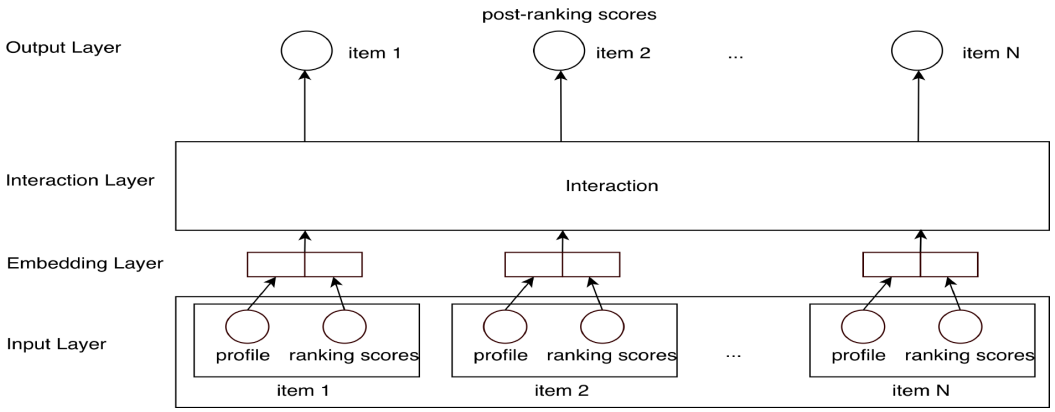

Fig. 6. The DNN architecture for post-ranking.

—Contextual features. They contain contextual information, such as time, location, and device type.
—Query. In search scenarios, the query feature is the user's textual input. In recommendation scenarios, this feature is often absent.
—Cross-features. They consist of cross-features representing interactions between different entities (e.g., between user profile and item profile, between user's historical behaviors and candidate item, between query and candidate item).

3 Overview of Deep Learning in Ranking Systems

In this section, we provide an overview of the deep learning techniques most commonly employed for industrial ranking systems.

Deep learning [96] is a subfield of machine learning that focuses on utilizing multi-layered neural networks to perform tasks such as classification, regression, and representation learning. It has achieved great success in numerous AI domains (e.g., NLP [43, 126], Computer Vision [70, 94], Speech Recognition [227]).

Popular deep learning techniques in industrial ranking include Embedding, Multi-MLP, Sequential Models, Multi-task Modeling, DRL, LLMs, and so on.

3.1 Embedding

Embedding techniques (e.g., Word2vec, Network embedding, Graph Neural Networks) learn to map discrete entities (words, items, users) into dense, continuous vector spaces that capture semantic relationships, providing rich inputs for downstream networks.

Word2vec. Word2vec [126], which learns distributed vector representations of words, is the foundation of modern deep learning in NLP.

Network Embedding. Network Embedding methods (e.g., DeepWalk [139], Node2vec [55], Struc2vec [152], NetMF [144]) extend the Word2vec paradigm to Network presentation learning, learning embedding vector representations of nodes in a network.

Graph Neural Networks. Graph Neural Networks (e.g., GCN [92], GAT [176], GraphSAGE [69]) have advanced graph modeling by learning node embeddings that incorporate both node features and the graph's topological structure (edges).

3.2 MLP

MLP [154] is a stack of fully connected layers with non-linear activation functions, forming a foundational feed-forward architecture for learning complex feature interactions.

3.3 Sequential Models

Sequential Models (e.g., RNN, Attention, Seq2Seq, Transformer) learn the temporal dependencies and patterns of sequential data.

Recurrent Neural Network (RNN). RNNs (e.g., LSTM [77], GRU [37]) utilize recurrent connections, where the output of a neuron at one timestep is fed back as input to the network at the next timestep. This mechanism allows RNNs to model temporal dependencies within sequences effectively.

Attention. Attention mechanism [7, 175] allows a model to dynamically focus on the most relevant parts of its input when making predictions, greatly improving performance on sequence tasks.

Seq2Seq. Seq2Seq [168] models use an encoder (e.g., an LSTM) to compress an input sequence into a context vector, and use a decoder to generate an output sequence.

Transformer. Transformer [57, 175] is a network architecture that relies on a multi-head attention mechanism and point-wise feed-forward networks, eliminating the need for recurrence and convolutions.

3.4 Multi-Task Modeling

Multi-task learning (e.g., **Multi-Gate Mixture-of-Experts (MMoE)**) seeks to construct a single model capable of learning multiple tasks concurrently.

MMoE. MMoE [119] adapts the Mixture-of-Experts framework to multi-task learning. It shares expert submodels across all tasks and employs a gating network trained to learn the weighting of these experts.

3.5 DRL

DRL is a machine learning subfield that integrates **Reinforcement Learning (RL)** with deep learning. It trains agents to make decisions by interacting with an environment to maximize cumulative reward, utilizing DNNs to represent state, policies, value functions, or environment models. Key approaches include Deep Q-learning, Policy Gradient, and Actor-Critic methods.

Q-Learning. Deep Q-network [128] approximates the optimal action-value function using a DNN and incorporates techniques such as experience replay and target networks to stabilize training.

Policy Gradient. Policy gradient methods (e.g., REINFORCE [188]) directly optimize the agent's policy by adjusting parameters in the direction that increases expected rewards.

Actor-Critic. Actor-critic algorithms (e.g., Asynchronous advantage actor-critic [127], Proximal Policy Optimization [156]) combine the strengths of value-based and policy-based methods. The actor updates the policy, while the critic evaluates the current policy using a value function.

3.6 LLMs

LLMs (e.g., BERT, GPT), which leverage large-scale Transformer-based models for pre-training and fine-tuning, have achieved remarkable performance. It has been one of the most exciting research fields in Deep learning. LLMs, which consist of Textual LLMs [9, 146, 166] and **Multi-Modal LLMs (MLLMs)** [2, 10, 145, 172], have demonstrated appealing scaling laws [91].

Textual LLMs. Textual LLMs [9, 146, 166] scale up Transformer-based autoregressive language models for text understanding, significantly enhancing task-agnostic and few-shot performance in NLP tasks. Some popular textual LLMs include:

- *GPT*. The GPT series (GPT [146], GPT-2 [147], GPT-3 [15], GPT-4 [2]) attains state-of-the-art performance in natural language understanding through generative pre-training on diverse unlabeled text corpora, followed by discriminative fine-tuning for specific tasks. They are widely used in generative chatbots like ChatGPT [131].
- *Qwen*. Qwen [9, 198] is an open source LLM developed by Alibaba.

MLLMs. A primary limitation of textual LLMs is their reliance solely on text features, which fail to capture multi-faceted information (e.g., images, audio) of items. To address this, MLLMs [2, 10, 145, 172] scale up Transformer-based models to process diverse data types such as text, images, audio, and video, enhancing their content understanding capabilities compared to textual LLMs. Prominent MLLMs include:

- *GPT-4*. GPT-4 [2] is a large-scale multi-modal model capable of accepting both image and text inputs.
- *Qwen-VL*. Qwen-VL [10] is Alibaba's MLLM-based on Qwen.
- *Gemini*. Gemini [172] is a family of highly capable MLLMs developed by Google.

Scaling Laws [91]. LLMs demonstrate compelling scaling laws, where model performance (e.g., cross-entropy loss) follows a power–law relationship with model size, dataset size, and computational resources used for training. Larger models are substantially more sample-efficient.

The subsequent sections provide detailed reviews of deep learning approaches for each stage of the Multi-stage Cascading Ranking System: matching (Section 4), pre-ranking (Section 5), fine-grained ranking (Section 6), post-ranking (Section 7), and relevance-ranking (Section 8).

4 Deep Learning Based Matching

This section introduces deep learning-based methods for matching, which is the first stage in a Multi-stage Cascading Ranking System. These methods are summarized in Table 1. We define the

Table 1. Deep Learning Based Methods for Matching

DL	LLM	Method Types	Models
No	No	Traditional Machine Learning	User-CF [109, 155], Item-CF [109], MF [93], Swing [202], Sceptre [122]
Yes	No	Embedding	Item2vec [11], Airbnb [54], EGES [180], GATNE [20]
Yes	No	Two-tower	Two-tower [210], DSSM [80], Amazon [130], DSPR [222], TwinBERT [116], DAT [213]
Yes	No	Multi-interest	MIND [98], ComiRec [19], Trinity [196], KuaiFormer [110]
Yes	No	Multi-objective	MOPPR [238], MOBIUS [44], VQ [14]
Yes	No	Graph Neural Network	PinSage [212], DecGCN [114], IntentGC [232], MEIRec [45]
Yes	No	Beyond Vector Dot Product	NANN [31], PDN [101], TDM [244], JTM [242], OTM [247], DR [52]
Yes	No	DRL	Top-K off-Policy [28]
Yes	Yes	Generative Retrieval	GR [215], TIGER [148], COBRA [204], GRAM [135], PinRec [6]
Yes	Yes	LLM Encoder-Based Retrieval	LEARN [86], ERNIE [115], NoteLLM [219, 220], LARM [113], PLUM [71]

matching problem in Section 4.1. Section 4.2 then reviews traditional matching methods widely used before the adoption of deep learning. Section 4.3 presents deep learning-based matching methods. Section 4.4 highlights matching approaches that leverage LLMs. Finally, Section 4.5 discusses advanced topics in matching, such as negative sampling and **Approximate Nearest-Neighbor (ANN)** search.

4.1 Definition of the Matching Problem

The matching (or candidate generation) stage aims to retrieve a subset of N_1 items (e.g., $N_1 = 10^4$) from the entire item corpus of size N (e.g., $N = 10^9$).

This stage presents several key challenges:

- Quality. The performance of the matching module sets the upper bound for the entire ranking system. Retrieved items should exhibit the highest possible relevance to the user.
- Scalability. The module must retrieve items from a corpus of billions within strict latency constraints.
- Completeness. It should retrieve diverse candidate types that a user may find interesting, including items similar to recently interacted ones, items aligned with long-term interests, popular items, and others.
- Diversity. Retrieved items should be diverse because an excess of similar items can reduce user engagement.

To address these challenges, the Industrial matching modules typically employ multiple matching methods simultaneously and merge their candidate outputs to form the final matching result.

4.2 Traditional Methods for Matching

This section introduces traditional matching methods that preceded deep learning and laid the foundation for subsequent deep learning-based matching approaches.

4.2.1 Traditional Matching Methods in Recommendation. Traditional recommendation matching methods include Collaborative filtering [109, 155], MF [93], and Graph-based methods [122, 202].

Collaborative Filtering [109, 155]. Collaborative filtering is one of the most successful matching approaches in recommender systems, providing recommendations based on the preferences of like-minded users. It consists of **User-Based Collaborative Filtering (User-CF)** and **Item-Based Collaborative Filtering (Item-CF)**.

- *User-CF* [109, 155]. User-CF identifies users similar to a target user, aggregates items from these similar users, and recommends them to the target user. Each user is represented as a

vector over items, with positive values for liked items and negative for disliked ones. Similarity between users is typically computed using cosine similarity.
— *Item-CF* [109]. Item-CF finds items similar to those a user has rated positively and combines them into a recommendation list. Item similarity is often calculated using cosine-based metrics [155]. This method allows efficient online recommendation through precomputed similar-item tables and scales independently of the number of users.

MF [93]. MF maps users and items into a shared latent factor space of dimension k based on the user–item rating matrix R. The rating matrix $\mathcal{R}$ is factorized into user and item latent matrices $\mathcal{P}$ and $\mathcal{Q}$. A user u's predicted rating on item i is given by $\hat{r}_{ui} = p_u^T q_i$.

Graph-Based Methods [122, 202]. Graph-based methods model relationships among items as a graph and use link prediction for item-to-item matching. For example, Sceptre [122] constructs a product graph with substitute and complementary relationships to recommend substitutable and complementary products in Amazon. Swing [202] builds a user–item click graph to predict substitutable products in Alibaba for Recommendation.

4.2.2 Traditional Matching Methods in Search Engines. Traditional search engine matching relies primarily on Inverted indexes [195] and Query rewriting [75].

Inverted Index [195]. Inverted index is a classical matching technique. In the offline stage, it tokenizes item text (e.g., titles or content) into terms, and builds an index mapping terms to items. In the online serving stage, the query is tokenized, and the index retrieves items associated with each query term.

Query Rewriting [75]. Query Rewriting addresses cases where short- or long-tail queries return insufficient results. It rewrites the original query into a similar form to better bridge the lexical gap between queries and items.

4.2.3 Problems of Traditional Matching Methods. Traditional methods suffer from several limitations.

Problems of Traditional Matching Methods in Recommendation. Traditional matching methods in recommendation have the following problems:

— They (e.g., User-CF [109, 155], Item-CF [109], MF [93], Swing [202]) typically rely only on ID features (i.e., user ID, item ID), ignoring rich content features (e.g., items' textual descriptions) that are valuable for recommendation.
— They (e.g., User-CF [109, 155], Swing [202]) often fail to model sequential patterns in users' historical behaviors or capture evolving user interests over time.

Problems of Traditional Matching Methods in Search. Traditional matching methods in search engines have the following problems:

— They cannot capture the semantic relationships between queries and items, relying solely on lexical matching. Semantically relevant items that do not contain exact query terms are missed.
— They lack personalization, unable to incorporate user characteristics (e.g., demographics) or behavior history. For example, a query for "shoes" should return different results for male and female users.

4.3 Deep Learning for Matching

To overcome the limitations of traditional methods, Deep learning-based matching methods [210] have been proposed, achieving state-of-the-art performance.

As shown in Table 1 ("DL" represents Deep Learning), these methods include Embedding, Two-Tower-based, Multi-interest, Multi-objective, Graph Neural Networks, Beyond Vector Dot Product, and DRL-based matching.

During training, these methods learn embeddings for users and items using rich feature sets (user profiles and historical behaviors, queries, item features). During serving, for a given user, they compute a vector representation of the user based on her features and typically retrieve top-ranked items via nearest neighbor search in the dot product space.

4.3.1 Embedding-Based Matching. Embedding-based matching approaches leverage word2vec [11, 54] or network embedding [20, 180] to learn item embeddings and identify similar items for matching.

Word2vec [126] has achieved great success in NLP by learning distributed representations of words. Inspired by this, Word2vec-based matching methods (e.g., Item2vec [11], Airbnb embedding [54]) apply word embedding algorithms to learn item embeddings and infer item–item relationships.

Network Embedding-based methods (e.g., DeepWalk [139], Node2vec [55], Struc2vec [152], NetMF [144]) extend the word2vec idea to network structures, learning node embeddings within the network. Network Embedding-based matching methods (e.g., EGES [180], GATNE [20]) frame matching as a link prediction problem in networks. EGES [180] constructs an item graph from user behaviors, learns item embeddings via DeepWalk [139], and incorporates side information to mitigate cold-start issues. GATNE [20] is a heterogeneous network embedding framework that models the multiple relationship types and item attributes.

4.3.2 Two-Tower-Based Matching. Embedding-based matching methods often rely solely on ID features (i.e., user id, item id) and fail to capture interest evolution within user behavior sequences. To address this, as shown in Figure 3, the Two-tower-based matching method [38, 80, 130, 210, 222] employs the user tower and the item tower to learn the representations of the user and item independently and computes the matching score between the user and the item via dot product. The user tower ingests features such as user profiles and historical behaviors, while the item tower processes item content features. During training, it learns the matching model. During inference, it precomputes items' embedding vectors, computes user embedding, and performs fast nearest neighbor search over the item embeddings. Unlike existing factorization methods (e.g., MF [93], Word2vec [126]) that use only IDs of users and items, the Two-tower-based method can incorporate arbitrary categorical and continuous features, leading to superior performance.

The Two-tower based matching method has seen widespread industrial adoption in industrial applications, including Video recommendation (YouTube [210]), E-commerce recommendation (Alibaba's SDM [117]), Web Search (Microsoft's DSSM [80]), E-commerce search (i.e., Amazon [130], JD's DSPR [222], Alibaba's MGDSPR [105]), and Search Ads (e.g., Baidu's MOBIUS [44], Microsoft's TwinBERT [116]), making it the prevailing matching method.

Two-tower-based method formulates candidate generation as extreme multi-class classification [210]. For a user u, the probability of selecting item i from corpus $\mathcal{I}$ follows a softmax distribution:

$$p(i|u) = \frac{e^{s(u,v_i)}}{\sum_{v_j \in \mathcal{I}} e^{s(u,v_j)}}, \qquad s(u, v_j) = u \cdot v_j, \tag{5}$$

where u and v_j denote user embedding and item embedding, respectively, $s(u, v_j)$ is the dot–product-based matching score between user u and item v_j. Given the large size of $|\mathcal{I}|$, the Two-tower-based method uses in-batch items as sampled negatives in computing the probability. The loss function is

the log-likelihood function:

$$L = -\frac{1}{N}\sum_{k=1}^{N} logp(i|u), \tag{6}$$

where N is the number of training instances.

After training, item embeddings are indexed for retrieval. During serving, the user embedding is computed via the user tower, and ANN search is performed over the item embedding index using tools like FAISS [88].

To enrich the interaction between user and item embeddings, extensions like TwinBERT [116] combine embeddings of user and item via a max operator and feed them into a residual network, while DAT [213] employs an augmented vector for each query and item.

4.3.3 Multi-Interest Matching. Two-tower-based methods typically represent a user with a single vector, which may not capture the user's diverse interests. To overcome this problem, Multi-interest matching methods [98, 110, 196] are proposed. MIND [98] extracts multiple interest vectors from user's behaviors via a dynamic routing mechanism. ComiRec [19] uses a multi-interest module to capture user's diverse interests from user behavior sequences, retrieves candidates using multiple interest vectors, and employs a controllable aggregation module to balance accuracy and diversity. Trinity [196] is a unified framework for optimizing multiple, long-tail, long-term interests for short-video matching. KuaiFormer [110] uses a Transformer to model long-term user sequences and has been deployed for matching in short-video recommendations, leveraging multiple query tokens to extract multi-interest representations.

4.3.4 Multi-Objective Matching. Information systems often involve multiple objectives (e.g., clicks, orders). Multi-objective matching methods [14, 44, 238] address this need. MOPPR [238] uses a weighted sum of losses for multiple objectives (i.e., exposure, relevance, click, order) in E-commerce. MOBIUS [44] integrates a relevance-ranking model to ensure retrieved items are relevant to the user's query in search ads. VQ [14] optimizes for multi-objective (e.g., stay-time, finish, interaction) in Douyin's short-video recommendation. Current multi-objective matching methods [14, 238] mainly use a shared-bottom architecture [119] with weighted loss optimization.

4.3.5 Graph Neural Networks-Based Matching. Graph Neural Networks-based matching methods (e.g., PinSage [212], DecGCN [114], IntentGC [232], MEIRec [45], LightGCN [72]) construct the graph of items, and learn items' embeddings via Graph Neural Networks for matching [47]. Graph Neural Networks (e.g., GCN [92], GAT [176], GraphSAGE [69], HGAT [32]) excel at modeling graph-structured data by jointly capturing node features and edge information. PinSage [212] learns item embeddings via Graph Convolutional Network for recommendation in Pinterest. For a query item, it recommends items whose embeddings are the K-nearest neighbors of the query item's embedding. To tackle the heterogeneous relationships among items, DecGCN [114] learns decoupled embeddings to infer substitutable and complementary items.

4.3.6 Beyond Vector Dot Product-Based Matching. Two-tower models [210], which rely on vector dot products with ANNs or **Maximum Inner Product Search (MIPS)** algorithms, have two key limitations: (1) the inner product may not capture complex user–item interactions; (2) ANN or MIPS is designed to approximate the learned inner product model, not directly optimized for the user–item interaction data. To address these, Beyond Vector Dot Product-based matching methods [31, 101, 244] have been proposed. Tree-based models (e.g., TDM [244], JTM [242], OTM [247]) offer logarithmic complexity relative to corpus size, even with expressive deep networks. NANN [31] extends ANN search to arbitrary matching functions, such as DNNs. Deep retrieval [52] learns a

retrievable structure directly from user–item interaction data without Euclidean space assumptions in ANN algorithms. PDN [101] combines embedding-based retrieval and item-to-item collaborative filtering via a Path-based Deep Network.

4.3.7 DRL-Based Matching. Matching models usually optimize for short-term objectives (e.g., click, purchase, watch time) [210]. To optimize long-term goals, DRL-based matching methods [28] have been introduced. DRL has achieved state-of-the-art results in long-term reward optimization, with successes in gaming(e.g., AlphaGo [164]), biological sciences (e.g., AlphaFold [90]), and so on. Some research applies DRL-based matching [28] to optimize long-term user engagement, which is critical in information systems. For example, Top-K off-Policy [28] uses a policy-gradient-based algorithm called REINFORCE for matching in YouTube's recommender systems.

4.4 LLM-Based Matching

Recently, LLM-based approaches have emerged for matching tasks, demonstrating promising results. These methods can be categorized into two main paradigms: Generative Retrieval and LLM encoder-based methods.

4.4.1 Generative Retrieval-Based Methods. Generative Retrieval-based methods [6, 135, 148, 204, 215] treat retrieval as a sequence generation problem by directly decoding the identifiers of relevant candidates.

GR [215] reformulates retrieval as a sequential transduction task, employing a **Hierarchical Sequential Transduction Unit (HSTU)** to encode users' long sequential behavior for generative retrieval within Meta's recommender systems. TIGER [148], an autoregressive generative retrieval approach, decodes target candidate identifiers. It utilizes a pre-trained LLM to generate dense item embeddings and applies a quantization scheme to create semantically meaningful tuples of codewords, which serve as a Semantic ID for each item. Given the Semantic IDs for items in a user session, a Transformer-based sequence-to-sequence model is trained to predict the Semantic ID of the next item with which the user will interact. This approach differs fundamentally from embedding-based retrieval methods, such as two-tower matching, which rely on a user interest embedding to retrieve items. COBRA [204], a generative recommendation framework that integrates sparse Semantic IDs and dense representations to improve accuracy and diversity, has been deployed in Baidu's Recommendation Ads. GRAM [135], which uses an LLM to generate text identifier codes for queries and items, is employed in JD's E-commerce Search retrieval. PinRec [6] is an outcome-conditioned, multi-token generative retrieval model designed for industry-scale Recommendation Systems at Pinterest. Its outcome-conditioned generation allows modelers to balance various metrics, such as saves and clicks, while multi-token generation enhances output diversity.

4.4.2 LLM Encoder-Based Retrieval Methods. LLM encoder-based retrieval methods leverage LLMs to learn item embeddings from content information (e.g., text, image, and video features) and use these embeddings for matching.

To align with the ranking system's characteristics, these methods typically perform **Supervised Fine-Tuning (SFT)** on a pre-trained LLM. The SFT process often involves multiple stages:

- Continued pre-training stage. These methods [115] first conduct continued pre-training on ranking system data to adapt to its distribution. For instance, ERNIE-based Retrieval [115] performs continued pre-training on query and document text from search engine logs.
- Target-specific fine-tuning stage. Subsequently, these methods [86, 219] fine-tune the model using standard training data for matching (e.g., predicting users' next clicked items). For example, ERNIE-based Retrieval [115] treats clicked items as positive examples. LEARN [86] employs an LLM as an item encoder and conducts retrieval-oriented fine-tuning within a

two-tower model, using the LLM's output as features for historically interacted items and target items to predict the user's next clicked item.

These fine-tuned LLMs excel at understanding item content. They enhance user understanding by first processing a user's historical interactions with an LLM to obtain item embeddings, then extracting user interest from this sequence of embeddings using sequential models like Transformers [71, 86].

Based on the modality they process, these methods are categorized into two types:

— Textual LLM-based methods. These methods use LLMs to learn item embeddings from textual features [86, 115, 219]. ERNIE-based Retrieval [115] employs a Transformer-based LLM to encode queries and items for Baidu's web search. LEARN [86] utilizes pre-trained LLMs as item encoders to better understand textual features in Kuaishou's short-video advertising recommendations. NoteLLM [219] applies a textual LLM to learn embeddings of items (i.e., notes) from their textual information for retrieval in Xiaohongshu's feed recommendation.
— MLLM-based methods. These methods [71, 113, 220] use MLLMs to learn item embeddings from features across modalities (e.g., text, image, and audio features), where video is typically processed as image frames. NoteLLM-v2 [220] uses an MLLM to extract visual information for retrieval. LARM [113] leverages MLLMs to extract video features for live-streaming recommendations on Kuaishou. PLUM [71] employs a MLLM to extract video features and represent items as semantic IDs for generative retrieval in YouTube's video recommendation.

LLM encoder-based retrieval methods offer two main advantages: (1) They improve the utilization of item content features for matching; (2) They are particularly effective for cold-start items with little user feedback, as LLMs can capture their characteristics from content features.

4.5 Advanced Topics in Matching

This section introduces advanced topics in matching: Negative sampling and ANN Search.

4.5.1 Negative Sampling. In training matching models, positive examples are typically items with which users have positively interacted (e.g., clicked, ordered). However, defining negative examples is a non-trivial challenge. Negative sampling is crucial for learning Two-tower-based retrieval models [79, 200, 210, 216]. The Two-tower approach [210] often uses in-batch items as negative samples. However, sampling batch negatives only from training data can result in models with poor discriminative power for long-tail items rarely seen during training. To address this, MNS [200] proposes a Mixed Negative Sampling approach, combining in-batch negatives with uniformly sampled negatives to mitigate the selection bias inherent in implicit feedback. Research in Facebook Search Retrieval [79] shows that using non-click impressions as negatives will lead to worse recall compared to random negatives, underscoring the importance of mining hard negatives. MGDSPR [105] generates relevance-improving hard negative samples in the embedding space. MOPPR [238] constructs multi-positive samples across the entire space for multi-objective retrieval models, utilizing impressed items, under-impressed items in multi-stage ranking and random negatives to optimize multiple objectives (i.e., exposure, relevance, click, purchase). MOBIUS [44] employs a relevance model to mine hard negatives for matching in search advertisements.

4.5.2 ANN Search. In the matching stage, ANN Search is typically performed on item embeddings to efficiently find a subset of items most similar to a user embedding, which is much faster than computing dot products exhaustively. Once the retrieval model is learned, embeddings of all items are generated by passing their features through the item tower. Libraries such as FAISS [88] are then used to build an index (e.g., HNSW [121], IVF-PQ [85]) over these item embeddings. This index

enables fast nearest neighbor search to retrieve the most similar items to the user embedding as matching results.

4.6 Future Work in Matching

Promising research directions in matching include LLM-based matching, Multi-objective matching, Beyond Vector Dot Product matching, and Negative Sampling strategies.

- —LLM-based matching. It represents a highly promising frontier for next-generation information retrieval systems. This approach encompasses two main techniques: Generative Retrieval and LLM encoder-based retrieval. For Generative Retrieval, prior work (e.g., GR [215], TIGER [148]) has demonstrated significant potential. Future research can advance this paradigm by: (1) Developing more sophisticated model architectures and refining training data design to enhance generative model performance. (2) Integrating generative and dense retrieval methods into synergistic approaches, as seen in models like COBRA [204]. (3) Improving the model's ability to satisfy multiple retrieval objectives simultaneously, as explored in PinRec [6].
  For LLM encoder-based retrieval, existing research (e.g., [86, 115, 219]) has validated the paradigm of using LLM for deep content understanding in dense retrieval. Key future directions include: (1) Advancing SFT methodologies tailored for retrieval tasks to learn better LLM embeddings (e.g., [86, 115, 219]). (2) Creating more effective techniques for integrating multi-modal embeddings (e.g., text, image, video) [71, 86, 113, 220] to achieve unified content understanding. (3) Improving the exploitation of learned textual or MLLM embeddings in downstream retrieval models (e.g., [86, 113]) to maximize their informational value.
- —Multi-objective matching. Multi-Objective-based Matching [14, 44, 238] is a critical and long-standing research direction because industrial ranking systems must often optimize multiple business objectives simultaneously. While current methods predominantly rely on the shared-bottom architecture [119], exploring more sophisticated multi-task learning techniques (e.g., MMoE [237], PLE [171]) holds significant promise.
- —Beyond Vector Dot Product matching. The performance of two-tower models is fundamentally limited by the simplicity of the vector dot product for scoring. Beyond Vector Dot Product-based matching methods (e.g., NANN [31], PDN [101]) address this limitation by enabling more expressive user–item interactions, thereby greatly improving matching performance. A promising future direction involves the continued exploration of novel, computationally efficient interaction mechanisms that can better capture complex relationships between users and items.
- —Negative Sampling. Negative examples play a critical role in learning effective matching models [79, 200, 210, 216]. Empirical studies have validated strategies like in-batch negative sampling [210] and mixing easy and hard negatives [200]. Moving forward, developing more sophisticated negative sampling strategies will be valuable for improving model discrimination and robustness.

5 Deep Learning-Based Pre-Ranking

This section introduces deep learning methods for pre-ranking, which follows matching in Multi-stage Cascading Ranking Systems. A summary of these methods is provided in Table 2. We first define the pre-ranking problem in Section 5.1. Subsequently, Section 5.2 details Deep learning-based methods (i.e., Two-Tower, and Two-Tower and MLP) for pre-ranking. Finally, Section 5.3 discusses prevalent optimization objectives (i.e., Multi-objective-based and Consistency-oriented) in Deep learning-based pre-ranking.

Table 2. Deep Learning-Based Methods for Pre-Ranking

Deep Learning	LLM	Optimization Objective	Methods Type	Models
Yes	No	Multi-objective-based	Two-Tower	ASMOL [228], HIT [199]
Yes	No	Multi-objective-based	DNNs	COLD [185]
Yes	No	Consistency-oriented	DNNs	COPR [236]

5.1 Definition of the Pre-Ranking Problem

The pre-ranking stage aims to select a subset of N_2 items (e.g., $N_2 = 10^3$) from the matching module's output (e.g., $N_1 = 10^4$ items). It handles a larger candidate pool under stricter latency constraints than the fine-grained ranker. The pre-ranking model should be a lightweight, efficient approximation of the more complex fine-grained ranking model.

Key challenges in pre-ranking include:

—Quality. The pre-ranking module's quality is critical, as it determines which items are passed to the subsequent fine-grained ranking stage.
—Scalability. The module must rapidly rank a lot of items (e.g., 10^4), a candidate set roughly 10 times larger than that handled by the fine-grained ranking stage.

5.2 Deep Learning for Pre-Ranking

As shown in Table 2, industrial pre-ranking architectures primarily follow two types: Two-Tower-based methods and Two-Tower and MLP-based methods.

5.2.1 Two-Tower-Based Methods. The Two-Tower-based pre-ranking model, similar to the Two-Tower matching model (Figure 3), uses the user tower and item tower to learn representations of users and items, and calculates the similarity between users and items via dot product. ASMOL [228] is a Two-Tower based pre-ranking model used in Taobao Search ($N_1 = 10^5$). HIT [199], a Hierarchical Interaction-Enhanced Two-Tower pre-ranking model that enriches the two-tower paradigm with coarse-grained user-ad interaction information, is deployed in Tencent Ads.

5.2.2 Two-Tower and MLP-Based Methods. As illustrated in Figure 4, the Two-Tower and MLP method uses the user tower and item tower to learn representations of users and items, then employs a MLP to model interactions between the user vector, item vector and their cross-features.

5.2.3 Comparison of Two-Tower and Two-Tower and MLP Pre-Ranking Model. The Two-Tower and MLP approach generally achieves better performance but requires more computational resources and incurs higher latency than the simpler Two-Tower model. The choice between these architectures often depends on the number of input items to the pre-ranking module. For a large input pool (e.g., $N_1 \geq 10^4$), the Two-Tower model is typically preferred to minimize computational cost and latency. For a smaller input set (e.g., $N_1 = 10^3$), Two-Tower and MLP-based methods (e.g., COLD [185]) are used to achieve better performance.

5.3 Optimization Objectives in Deep Learning-Based Pre-Ranking

In industrial pre-ranking systems, optimization objectives generally fall into two categories: Multi-objective models and Consistency-oriented cascade models.

5.3.1 Multi-Objective-Based and Consistency-Oriented Pre-Ranking Models.

—*Multi-objective-based models*. These approaches learn multiple objectives for pre-ranking directly from user feedback. For instance, COLD [185] treats clicked items as positive samples

and exposed-but-unclicked items as negative samples to learn a CTR pre-ranking model. ASMOL [228] utilizes clicked items, ordered items, exposed items, items in the pre-ranking stage, and items in the ranking stage to train a multi-objective (e.g., CTR, CVR, Relevance) pre-ranking model in Taobao Search.

— *Consistency-oriented cascade models*. These methods train the pre-ranking model to align with the ranking order of downstream stages (e.g., Fine-grained ranking stage), thereby enhancing consistency across the ranking cascade. COPR [236], a consistency-oriented pre-ranking framework, explicitly optimizes for alignment between pre-ranking and subsequent ranking results. Its optimization objective ensures that items ranked highly in the fine-grained ranking stage receive higher pre-ranking scores than those ranked lower. This framework has been deployed in Taobao's recommendation advertising system.

5.3.2 Comparison of Multi-Objective-Based and Consistency-Oriented Pre-Ranking Models. Consistency-oriented cascade models typically achieve higher passing rates and performance metrics than multi-objective models, as they directly learn from the ranking order of downstream stages. However, their performance can be constrained by the quality of the downstream ranking results. Conversely, the multi-objective model benefits from directly using user feedback as ground truth, free from the limitations imposed by downstream ranking orders.

5.3.3 Combination of Multi-Objective-Based and Consistency-Oriented Pre-Ranking Models. Consequently, industrial pre-ranking modules often combine both optimization approaches. A common deployment strategy employs parallel models, where the multi-objective and consistency-oriented models are treated as separate entities. These two models require distinct training data and are developed independently during the offline training. For online serving, they are deployed as independent services. Upon receiving a user request, the two models generate their respective pre-ranking scores, s_{i_0} and s_{i_1}, in parallel. Then, the final pre-ranking score for item i is computed by combining s_{i_0} (from the Multi-objective-based model) and s_{i_1} (from the Consistency-oriented cascade model), as defined in Equation (7):

$$final\_pre\_ranking\_score = (1 + \alpha_0 * s_{i_0})^{\beta_0} * (1 + \alpha_1 * s_{i_1})^{\beta_1}, \tag{7}$$

where $\alpha_0, \alpha_1, \beta_0, \beta_1$ are tunable weight parameters.

5.4 Future Work in Pre-Ranking

Promising research directions for pre-ranking include: Developing more sophisticated pre-ranking models and Consistency-oriented pre-ranking models.

— Sophisticated pre-ranking models. Current industrial pre-ranking models primarily rely on the Two-Tower or Two-Tower and MLP paradigms [185, 228] for efficiency. However, this architecture limits expressive power by preventing deep feature interactions between users and items. A promising research direction is to explore more powerful yet efficient architectures, such as those incorporating lightweight attention mechanisms between items and user behavior sequences or introducing critical feature interactions. This would narrow the performance gap with fine-grained ranking while maintaining efficiency.

— Consistency-oriented pre-ranking. Research has shown that improving consistency between Pre-ranking and Fine-grained ranking leads to significantly better pre-ranking performance [236]. The future of this direction lies in exploring more advanced consistency-learning techniques, such as developing better methods to leverage relative ranking orders information from the fine-grained ranker.

Table 3. Deep Learning-Based Methods for Fine-Grained Ranking

DL	LLM	Method Types	Models
No	No	Traditional Machine Learning	LR [124], GBDT [211], LR and GBDT [74], FM [151], FFM [89]
Yes	No	DNNs	YouTube DNN [38], Airbnb DNN [67, 68], Wide and Deep [36], DLRM [129]
Yes	No	Sequence Modeling	DIN [240], DMT [60], BST [30], TransAct [191], DIEN [239], RACP [48], FIM [178]
Yes	No	Long Sequence Modeling	SIM [142], ETA [29], SDIM [17], QIN [65], TWIN [22], TWIN V2 [162], TransAct-v2 [192]
Yes	No	Feature Interactions	DCN [181], DCN-v2 [182], GDCN [177], DeepFM [64], AutoInt [165], CAN [13]
Yes	No	Multi-task Learning	MMoE [237], DMT [60], PLE [171], MoSE [143], ESSM [120], HoME [183], AdaTT [100], STEM [134]
Yes	No	Multi-domain	STAR [160], ZEUS [58], PEPNet [23], APG [194], POSO [39], HoME [183], MLORA [205]
Yes	No	Multi-modal	Image CTR [53]
Yes	Yes	Generative Ranking	GR [215], GenRank [81], LUM [193], GPR [223], OneRec [42], OneSug [66], OneLoc [186]
Yes	Yes	Sequence Modeling Scaling	LONGER [21], STCA [62], PinFM [33]
Yes	Yes	Feature Interactions Scaling	Wukong [217], DHEN [218], Hiformer [63], RankMixer [245], OneTrans [229], HyFormer [82]
Yes	Yes	LLM Encoder	LEARN [86], ERNIE [115], HLLM [26], SimTier and MAKE [159], MUSE [189], MOON [221], LARM [113]

6 Deep Learning-Based Fine-Grained Ranking

This section introduces deep learning-based methods for Fine-grained ranking (often called "Ranking"), which follows pre-ranking in a Multi-stage Cascading Ranking System. These methods are summarized in Table 3. We define the fine-grained ranking problem in Section 6.1. Section 6.2 then reviews traditional methods that were widely adopted prior to the use of deep learning. Building on this foundation, Section 6.3 details deep learning-based methods for fine-grained ranking. Section 6.4 examines emerging techniques that leverage LLMs. Finally, Section 6.5 discusses advanced topics, including the Delay Feedback problem and loss function design.

6.1 Definition of the Fine-Grained Ranking Problem

The fine-grained ranking (or Ranking) stage retrieves a subset of N_3 items (e.g., $N_3 = 10^2$) from the pre-ranking module's output (e.g., $N_2 = 10^3$ items).

This stage presents two primary challenges:

— Quality. The ranking module's quality is critical, as only its selected items (e.g., 10^2) are passed to the subsequent post-ranking stage.
— Scalability. The module must select this subset from approximately 10^3 items with low latency.

6.2 Traditional Methods for Ranking

The Industrial ranking module predominantly used traditional machine learning models (e.g., LR [124], GBDT [60]) as ranking models prior to 2012.

LR [124] applies LR for CTR prediction in Google Search. GBDT has been widely used for both Search [211] and Recommendation [60]. The LR and GBDT [74] approach uses GBDT to generate features for an LR-based ranking model in Meta's recommendation Ads.

Feature Crossing. To enhance feature interactions more effectively, **Factorization Machine (FM)**-based methods [89, 151] were proposed. FM [151] models interactions between features using factorized parameters. FFM [89] extends this concept with field-aware FM for CTR prediction.

6.3 Deep Learning for Fine-Grained Ranking

Deep learning has achieved significant success by consistently outperforming traditional methods (e.g., LR [124], GBDT [74], FM [151]) for fine-grained ranking in industrial systems. As summarized in Table 3, these methods encompass DNNs, Sequential Modeling, Long Sequence Modeling, Feature Interaction Modeling, Multi-task learning, Multi-domain learning, and Multi-modal approaches.

6.3.1 DNNs-Based Methods. DNNs-based ranking models typically follow the Embedding and MLP paradigm [36, 38, 67, 73]. In this paradigm, input features are firstly mapped into low-dimensional embedding vectors, grouped and transformed into fixed-length vectors, concatenated, and then fed into an MLP to learn non-linear feature relations.

YouTube DNN [38] applied DNNs for ranking in YouTube's recommendation system. Deep Crossing [157] employed DNN to automatically combine features for Microsoft's Web Search. Airbnb DNN [67, 68] shared practical experiences in applying DNNs to Airbnb Search. Wide and Deep [36] jointly trains DNNs and wide linear models to combine the benefits of generalization and memorization for Google Play recommendations. DLRM [129] demonstrated the effectiveness of DNN in Meta's recommender systems. Google Search Ads [4] presented engineering insights for CTR prediction in industrial-scale advertising recommendation.

6.3.2 Sequential Modeling. Sequential models aim to extract user interest representations from historical behaviors, which is critical in industrial ranking systems. These methods include pooling, attention, RNN, Transformers, and frequency-aware approaches.

- *Pooling*. YouTube DNN [38] uses average pooling to aggregate a user's historical behaviors into a single vector.
- *Attention.* DIN [240] highlights that users may have multiple interests in historical behaviors, and uses attention mechanism to adaptively learn the representation of user interests which varies over different target items.
- *RNN* [76, 103, 149, 239]. To model the evolution of user interests over time, RNN-based approaches were proposed. DIEN [239] incorporates an RNN-based interest evolving layer to capture dynamic user interest processes. HUP [59] uses LSTM to model fine-grained micro behaviors (e.g., clicks on item components like pictures or comments) for E-commerce recommendation. DSIN [50] applies LSTM to model how users' interests evolve among sessions in Taobao's E-commerce recommendation. RACP [48] uses an RNN to model user's contextualized page-wise feedback within a session for Taobao search.
- *Transformer* [30, 60, 166, 191]. Transformer-based approaches [30, 60, 191, 203] have achieved state-of-the-art performance in sequence modeling for ranking. DMT [60] employs multiple transformers to simultaneously model different types of user behavior sequences in JD's E-commerce recommender system. BST [30] utilizes a self-attention-based architecture in Taobao's recommender system. TransAct [191] employs a Transformer to model user's real-time behavior sequences for Pinterest recommendations.
- *Frequency-aware* [178, 241]. These techniques process user behavior in the frequency domain. FMLP-Rec [241] is an all-MLP model that applies learnable filters to reduce noise in historical behavior data in the frequency domain. FIM [178] uses the Fourier transform to convert temporal behavior sequences into the frequency domain, allowing the model to dynamically perceive users' periodic patterns.

6.3.3 Long Sequence Modeling. Modeling user interests from long-term behavior sequences is crucial in industrial ranking systems [29, 140, 142], as these long-term behaviors faithfully reflect user preferences. SIM [142] employs a **General Search Unit (GSU)** to extract a subset of user behavior sequences relevant to candidate items from long sequential data, followed by an **Exact Search Unit (ESU)** to model precise relationship between candidate items and subsequences in advertising recommendations. ETA [29] utilizes locality-sensitive hashing to significantly reduce the costs of identifying similar items in long sequence modeling. SDIM [17] proposes a sampling-based end-to-end approach for modeling long-term user behaviors by sampling multiple hash functions to generate hash signatures for both candidate items and items in the user behavior sequence. QIN [65]

first employs a relevance search unit to identify subsequences related to user queries and then further extracts sub-subsequences relevant to target items in Kuaishou search. TransAct-v2 [192] models lifelong user action sequences for Pinterest recommendations. However, these methods face a key limitation: inconsistent relevance metrics between the target item and user behaviors in the GSU and ESU stages. To address this, TWIN [22] introduces a consistency-preserving approach that aligns GSU and ESU objectives. TWIN V2 [162] further employs hierarchical clustering to group ultra-long behavior sequences, enabling more accurate and diverse user interest extraction.

6.3.4 Feature Interactions. Identifying effective feature interactions is essential for learning ranking models [36, 182]. The following categories of feature interaction models have been proposed:

- *DNN-based*. Deep Crossing [157] exploits MLP to automatically learn implicit cross-feature representations.
- *Explicitly feature crossing-based*. Wide and Deep [36] combines wide linear models with DNNs to explicitly model feature interactions for Google Play recommendations. MemoNet [224] uses a multi-hash codebook network as a memory mechanism to efficiently learn and store cross-feature representations in CTR prediction.
- *FM-based*. Inspired by FM [151], DeepFM [64], and NFM [73] integrate FM with DNNs for ranking tasks.
- *DCN-based*. DCN [181] explicitly crosses features at each layer with input features. DCN-v2 [182] enhances DCN's expressiveness in modeling complex explicit cross-features and is deployed in Google's recommender systems. GDCN [177] captures high-order feature interactions and dynamically filters important interactions via an information gate.
- *Transformer-based*. AutoInt [165] uses self-attention mechanisms to automatically learn high-order feature interactions of input features.
- *Parameter Network-based*. CAN [13] uses a Co-Action Network to approximate explicit pairwise feature interactions without introducing excessive parameters, computing interactions by passing one feature's embedding through another feature's parameter network (i.e., MLP).

6.3.5 Multi-Task Learning. Multi-task learning-based methods aim to optimize multiple objectives in industrial ranking systems simultaneously. YouTube MMoE [237] applies MMoE [119] to optimize for multiple ranking goals in YouTube recommendations. DMT [60] uses MMoE to simultaneously optimize multiple tasks in JD's E-commerce recommender systems. PLE [171] explicitly separates shared and task-specific experts and employs a progressive routing mechanism to extract and disentangle deeper semantic knowledge gradually in Tencent's News recommendation. HoME [183] utilizes a hierarchy of multi-gate experts for multi-task learning in Kuaishou's short-video recommendation. AdaTT [100] constructs a deep fusion network with task-specific and shared fusion units at multiple levels in Meta's recommender systems. ESSM [120] models CVR prediction directly over the exposure space by jointly modeling CTR and CVR tasks. STEM [134] incorporates task-specific embeddings to capture user's varying interests across tasks.

6.3.6 Multi-Domain Learning. Industrial information systems often have multiple domains and scenarios with varying data distributions. Multi-domain learning models [23, 58, 160, 194] aim to capture domain-specific characteristics and can be categorized into Domain-wise and Instance-wise methods.

- *Domain-wise methods*. They learn a specialized model for each domain. STAR [160] is a single model that serves all domains, where the network of each domain consists of two factorized networks: one centered network shared by all domains and one domain-specific network tailored for each domain. ZEUS [58] first performs self-supervised learning on users'

spontaneous behaviors to mitigate the feedback loop problem in Recommender systems and generates a pre-trained model, and then learns domain-aware ranking models for multiple scenarios by fine-tuning the pre-trained model based on users' implicit feedback in multiple scenarios. POSO [39] introduces user-group-specialized sub-modules to enhance personalization in ranking models. MLORA [205] proposes a Multi-domain Low-Rank Adaptive network for CTR prediction, incorporating a specialized LoRA [78] module for each domain. HoME [183] uses a hierarchy masking mechanism to improve sharing efficiency between tasks and gating mechanisms to ensure each expert could obtain appropriate gradient to maximize its effectiveness.

— *Instance-wise methods*. They learn specific neural network parameters for each sample instance. Inspired by LHUC [169], PEPNet [23] incorporates personalized prior information to dynamically scale embeddings and DNN hidden units via a gate mechanism. APG [194] dynamically generates network parameters based on instance-level features.

6.3.7 Multi-Modal. Modeling multi-modal data [53, 159] is critical in large-scale industrial ranking systems because the multi-modal features can reflect both item characteristics and user preferences. For example, Image CTR [53] captures user image preferences by modeling images of target items and user's historical items in Taobao's E-commerce advertising recommendation.

6.4 LLM-Based Fine-Grained Ranking

Recently, LLMs-based methods [215] have achieved great success in fine-grained ranking. They consist of Generative Ranking, Sequence Modeling Scaling, Feature Interactions Scaling, and LLM encoder-based methods.

6.4.1 Generative Ranking. Generative Ranking methods [42, 66, 81, 193, 215] directly decode target candidates' identifiers. GR [215] reformulates ranking as a sequential transduction task, employing a HSTU to encode user's long sequence for generative ranking in Meta's recommender systems. GenRank [81] treats items as positional information and iteratively predicts user behaviors. LUM [193] introduces a "next-condition-item prediction" task to align generative pre-training with discriminative applications. Recently, the One-Model paradigm (e.g., OneRec [42], OneSug [66], OneLoc [186], OneSearch [25], GPR [223]) proposes end-to-end generative ranking models that directly generate a list of items, replacing the multi-stage cascading ranking paradigm to address issues like computational fragmentation and optimization inconsistency problems. OneRec [42] encodes the user's historical behavior sequences and gradually decodes the videos that the user may be interested in Kuaishou's video recommendation. OneSug [66] offers a unified End-to-End generative framework for query suggestions. OneLoc [186] provides geo-aware generative recommendations for local services in Kuaishou. OneSearch [25] is an end-to-end generative framework for e-commerce search in Kuaishou. GPR [223] is a generative pre-trained one-model paradigm for large-scale advertising recommendations in Tencent.

6.4.2 Sequence Modeling Scaling. Inspired by scaling laws [91] and Transformer success in LLMs, Sequence Modeling Scaling-based methods [21, 33, 62] scale up Transformers for long sequence modeling. LONGER [21] optimizes transformer for GPU-efficient recommenders in Bytedance by compressing sequences of length 2,000 via token merging. STCA [62] uses Stacked Target-to-History Cross Attention to directly model sequences up to 100,000 in length for ranking in Douyin's Short-Video recommender systems. PinFM [33] pre-trains a 20B+ parameter transformer foundation model on extensive user activity data and fine-tunes it for billion-scale visual discovery at Pinterest.

6.4.3 Feature Interactions Scaling. Feature Interactions Scaling-based methods [82, 217, 229, 245] scale up feature interaction components to establish analogous scaling law for ranking systems, where the ranking quality can be continuously improved in conjunction with dataset size, compute power, and parameter budgets. Wukong [217] uses a network of stacked factorization-machine blocks and linear compression blocks to capture arbitrary-order feature interactions, demonstrating strong scaling properties within Meta's recommender systems. Similarly, DHEN [218] employs a deep hierarchical ensemble of heterogeneous interaction modules, learning a hierarchy of feature interactions across different orders, and achieves effective scaling by progressively stacking multiple feature-crossing layers. Hiformer [63] employs a heterogeneous Transformer for feature crossing in Google, enhancing expressiveness through heterogeneous self-attention (feature field aware parameters in query, key, value projections) and heterogeneous Feed-Forward Networks (feature field aware parameters in FFN). RankMixer [245] replaces Hiformer's heterogeneous self-attention module with the multi-head token mixing module (similar to Mlp-mixer [173]), and leverages a sparse Mixture of Experts in heterogeneous FFN (i.e., Per-token Feed-Forward Networks) to scale to one billion dense parameters. OneTrans [229] and HyFormer [82] utilize a unified Transformer backbone to concurrently handle user-behavior sequence modeling and high-order feature interaction learning.

6.4.4 LLM Encoder-Based Methods. They leverage LLMs to learn item embeddings from content information (e.g., text, images, and video features) and subsequently use these embeddings for ranking. These methods can be categorized into two types based on their input modalities:

- Textual LLM-based methods. These methods employ LLMs to extract item embeddings from textual features [26, 86, 251]. For example, ERNIE-based Ranking [251] uses a Transformer-based LLM to model the concatenation of query and item embeddings for Baidu's web search ranking. HLLM [26] uses an Item LLM to extract embeddings from item text descriptions and a User LLM to predict future user interests based on interaction history and the extracted item embeddings. LEARN [86] utilizes pretrained LLMs as item encoders to extract embeddings from textual features for Kuaishou's short-video recommendation.
- MLLM-based methods. These methods [113, 159] use MLLMs to extract embeddings from multi-modal features. For instance, SimTier and MAKE [159] employs pre-training on multi-modal item representations (i.e., image and text) to capture semantic similarity for product recommendation in E-commerce. MUSE [189] exploits multi-modal embeddings to retrieve related items from a user's long-term behavior sequence for sequence modeling in display advertising. MOON [221] is a generative MLLM-based model for product representation learning in e-commerce. LARM [113] leverages MLLMs to capture video features and model temporal dynamics for live-streaming recommendation in Kuaishou.

6.5 Advanced Topics in Fine-Grained Ranking

In this section, we introduce advanced topics in fine-grained ranking: The Delay Feedback problem and Loss function.

6.5.1 The Delay Feedback Problem. The Delay Feedback problem is common in industrial ranking systems, which update ranking models in an online streaming learning manner to catch up with the real-time data distribution, as training labels (e.g., conversion in E-commerce) may arrive hours or days after an impression. This creates a tradeoff between waiting for accurate labels and using fresh data: a short waiting window risks mislabeling positives as negatives, while a long window delays model updates. Several methods address this challenge. DFM [24] introduces an auxiliary model to capture conversion delay for CVR prediction in Criteo recommendation ads. Twitter [95] tackles

delayed feedback in CTR prediction with principled loss functions. FSIW [208] addresses the delay feedback problem in CVR prediction at Cyberagent using an importance weighting approach. ESDF [184] discretizes delay time into day slots and models the probability via survival analysis with DNN for CVR prediction. ES-DFM [201] models the relationship between observed and true conversion distributions, and estimates instance-specific importance weights for the loss function in Taobao Search. DEFER [56] incorporates real negative samples and uses importance sampling to adjust loss weights in CVR prediction at Taobao's recommendation ads. DEFUSE [35] refines importance weights at a finer granularity for different sample types (immediate positive, fake negative, real negative, and delay positive). Google [5] splits the label as a sum of labels with different delay buckets and trains each label bucket only on the mature label for CVR prediction.

6.5.2 Loss Function. Loss functions for learning ranking models are broadly categorized into three types:

- —Point-wise loss function [107, 111]. Industrial ranking systems usually treat the ranking problem as a binary classification problem and use the cross-entropy loss function to learn the ranking models (e.g., CTR, CVR).
- —Pairwise loss function. Studies [99, 107, 158] demonstrate the effectiveness of combining cross-entropy loss with pairwise ranking loss for CTR and CVR prediction.
- —Listwise loss function. Yan et al. [197] combine cross-entropy loss with pairwise or listwise ranking loss for CTR prediction in Google Search Ads. Similarly, Bai et al. [8] show the effectiveness of combining cross-entropy loss with list-wise ranking loss for CTR prediction in YouTube Search.

6.6 Future Work in Fine-Grained Ranking

Promising research directions for fine-grained ranking include: LLM-based Ranking, Long Sequence Modeling Scaling, Feature Interactions Scaling, Multi-task Learning, and Multi-domain Learning.

- —LLM-based ranking. LLM-based ranking represents a transformative innovation in information retrieval, consisting of Generative Ranking and LLM encoder-based ranking. For Generative Ranking, research has demonstrated its effectiveness in ranking [42, 66, 81, 193, 215]. Future directions include: (1) Developing more robust and efficient generative architectures specifically designed for ranking [215]. (2) Innovating in the design of high-quality training data to teach models to learn efficiently from user feedback [81]. (3) Pioneering methods to drastically reduce inference latency and computational cost, which are primary barriers to online deployment.
  For LLM encoder-based ranking, existing research has proven the effectiveness of using LLM as powerful content embedding extractors for ranking (e.g., [26, 86, 113, 159, 251]). Critical future directions include: (1) Advancing supervised fine-tuning techniques to produce more discriminative and task-specific embeddings from LLMs for ranking (e.g., [81, 193, 215]). (2) Designing novel model architectures that can more effectively incorporate and utilize these rich LLM-derived embeddings (e.g., [113, 159]).
- —Long sequence modeling. Effectively modeling users' long behavior sequences is paramount for capturing user intent in industrial ranking systems (e.g., SIM [140], ETA [142], TWIN V2 [162], LONGER [21], STCA [62]). The field has evolved from retrieval-based methods (e.g., hard search [140], locality-sensitive hashing [142], consistency-preserved retrieval [162]) to end-to-end modeling techniques [21, 62, 215]. Promising research directions include: (1) Developing novel models and engineering solutions that balance the expressive power of end-to-end modeling with its computational and infrastructural costs. (2) Pioneering methods to

Table 4. Deep Learning-Based Methods for Post-Ranking

Deep Learning	Method Types	Models
No	Traditional Machine Learning	MRR [18], DPP [27, 187]
Yes	List-wise	PRM [138], DLCM [3], Airbnb Search [1], RankFormer [16]
Yes	Generator–Evaluator	PRS [49], PIER [161], NAR4Rec [150], Seq2slate [12], MiRNN [246]
Yes	DRL	SlateQ [83], Value-based RL [137]

model a user's complete, lifelong behavioral history to achieve a truly holistic understanding of user preferences.

—Feature interactions. Capturing sophisticated feature interactions is a cornerstone of high-performing ranking models (e.g., DCN-v2 [182], Wukong [217], Hiformer [63], RankMixer [245]). A primary research direction involves investigating more expressive and efficient interaction mechanisms. The key challenge is to design models that can automatically identify high-order, non-linear interactions between features without incurring prohibitive computational overhead, thereby pushing the boundaries of predictive accuracy.

—Multi-task learning. Multi-task Learning (e.g., MMoE [237], PLE [171]) is of enduring importance as industrial ranking systems inherently involve optimizing multiple objectives. Promising research directions are: (1) Designing more sophisticated parameter-sharing and gating mechanisms to better model synergies and conflicts between tasks, leading to improvements across all objectives. (2) Enhancing the performance of a primary task (e.g., CVR prediction) by leveraging supervisory signals from auxiliary tasks (e.g., dwell time, add-to-cart).

—Multi-domain learning. Industrial ranking systems usually have multiple domains and scenarios with varying data distributions. Multi-domain learning (e.g., STAR [160], ZEUS [58], PEPNet [23]) is essential for building unified, scalable ranking models. A significant research direction is to develop more robust and parameter-efficient architectures that can effectively learn both shared knowledge and domain-specific characteristics, enabling strong performance across all scenarios while minimizing training and maintenance complexity.

7 Deep Learning-Based Post-Ranking

This section introduces deep learning methods for post-ranking, which follows fine-grained ranking in Multi-stage Cascading Ranking Systems, as summarized in Table 4. We define the post-ranking problem in Section 7.1. Section 7.2 then reviews traditional post-ranking methods that were prevalent before the adoption of deep learning. Finally, Section 7.3 presents Deep learning-based approaches for post-ranking.

7.1 Definition of the Post-Ranking Problem

The post-ranking stage selects top-K items (e.g., $K = 10$) from the output of the fine-grained ranking stage (e.g., $N_3 = 10^2$ items). These items are directly presented to the user and must account for both mutual influences among items and a balance between short-term efficiency metrics (e.g., click, conversion) and long-term engagement metrics (e.g., diversity, dwell time, and user retention). In contrast, the fine-grained ranking stage evaluates items independently without considering their interdependencies.

Key challenges in post-ranking include:

—Quality. The quality of the post-ranking module is critical because the selected items are directly exposed to users.

—Short-term and long-term engagement optimization. The module must optimize for both immediate user interactions (e.g., clicks) and long-term user engagement (e.g., dwell time and retention).

7.2 Traditional Methods in Post-Ranking

Traditional Methods (e.g., MRR [18] and DPP [27, 187]) in post-ranking primarily focus on diversification [18], which is an important objective in post-ranking, as presenting varied results offers users greater choice and improves overall experience.

MRR [18] is a diversification method that selects items based on marginal relevance, favoring documents that are both relevant to the query and minimally similar to previously selected items. DPP [27, 187] employs a determinantal point process to generate recommendations that are both relevant and diverse.

7.3 Deep Learning-Based Post-Ranking

Deep learning-based post-ranking methods [138] directly learn the post-ranking model based on user's feedback (e.g., clicks). As shown in Table 4, Deep learning-based post-ranking methods can be categorized into List-wise models, Generator–Evaluator methods, and DRL-based methods.

—*Listwise-based methods*. As illustrated in Figure 6, List-wise-based methods use an interaction layer such as self-attention [138] to capture the influence among items. PRM [138] employs a Transformer to model the mutual influence between items for post-ranking in E-commerce recommendation. DLCM [3] uses a RNN to encode the top retrieved results and refines the ranked list through a local context model. Airbnb Search [1] applies a RNNs-based model to diversify search results in Airbnb. RankFormer [16] leverages Transformer to learn from both relative feedback on individual items and absolute user feedback on the overall list.

—*Generator–Evaluator-based methods*. The Generator–Evaluator architecture operates in two stages: first, heuristic methods such as beam-search generate several top-K candidate lists; second, an evaluation model selects the optimal permutation from the candidate lists. Seq2slate [12] is a sequence-to-sequence model for post-ranking that predicts the next best item given the already selected items. MiRNN [246] frames post-ranking in E-commerce search as a sequence generation problem, using a RNN to capture the mutual influences between items and optimize the ranking order for GMV metric. PRS [49] is a permutation-wise two-stage framework consisting of a permutation-matching stage and a permutation-ranking stage. In the permutation-matching stage, it uses permutation-wise and goal-oriented beam search to generate multiple candidate lists. In the permutation-ranking stage, it employs a permutation-wise ranking model to evaluate the candidate lists. PIER [161] integrates the generator and evaluator into an end-to-end trainable model. NAR4Rec [150] adopts a non-autoregressive approach for post-ranking, using a matching model to improve convergence and a sequence-level training method to enhance overall utility.

—*DRL-based methods*. DRL-based methods [83, 137, 234, 249] directly optimize the desire metrics-based on DRL. SlateQ [83] decomposes the long-term value of a rank list into a tractable function of its constituent items in YouTube recommendations. Value-based RL [137] applies an evolution-strategy-based RL algorithm to directly optimize the overall value of the ranking list.

7.4 Future Work in Post-Ranking

The promising research direction for post-ranking is: Long-term reward optimization.

Table 5. Deep Learning-Based Methods for Relevance-Ranking Model

Deep Learning	LLM	Method Types	Models
No	No	Traditional Machine Learning	GBDT [211]
Yes	No	Embedding and MLP	DSSM [80], MASM [206]
Yes	No	BERT-based	ReprBERT [207], SPM [214], DeepBoW [108]
Yes	Yes	LLM-based	ELLM-rele [231], GenFR [209], HCMRM [51], Walmart [125]

—Long-term reward optimization. While post-ranking has traditionally focused on optimizing diversity (e.g., MRR [18]) or item relationships (e.g., PRM [138]), the most critical yet challenging research direction is the direct optimization of long-term user value (e.g., user's retention), which is paramount to the health of an online platform. Although some research (e.g., FeedRec [248]) has attempted to use DRL for this problem, these methods face significant practical barriers to industrial adoption. Consequently, developing practical and scalable methods to directly optimize long-term rewards in post-ranking remains a significant and open research problem.

8 Deep Learning-Based Relevance-Ranking

This section details deep learning methods for Relevance-ranking, which is a core task in search. Relevance-ranking models are often used in the matching, pre-ranking, and fine-grained ranking stages of industrial search engines and search advertising. A taxonomy of these models is provided in Table 5. Section 8.1 establishes the definition of the Relevance-ranking problem. Section 8.2 reviews traditional methods that preceded deep learning. The core of our discussion in Section 8.3 focuses on Deep Learning-based methods, leading to a dedicated analysis of cutting-edge LLM-based techniques in Section 8.4.

8.1 Definition of the Relevance-Ranking Problem

Industrial search engines and search advertising systems employ Relevance-ranking models [80, 206, 211] to ensure that the retrieved items are relevant to the user's query. The relevance models compute the relevance scores between queries and retrieved items.

Relevance-Ranking in Matching, Pre-Ranking, and Fine-Grained Ranking Stages. Relevance-ranking models of varying computational complexity are typically deployed across matching, pre-ranking, and fine-grained ranking stages to ensure the final results are relevant to the query. During matching and pre-ranking, a lightweight relevance model filters out less relevant results (e.g., items not belonging to the query's category). In the fine-grained ranking stage, a sophisticated relevance model produces more accurate relevance scores, which prioritize the most relevant results.

Challenges of Relevance-Ranking. Relevance-ranking faces two primary challenges:

—Semantic matching. Queries and items may be semantically related despite lacking lexical overlap.
—Limited training data. Establishing ground-truth relevance requires costly human-labeled relevance data.

8.2 Traditional Methods in Relevance Model

Traditional relevance models typically utilize classical machine learning techniques such as GBDT [211] to generate relevance scores.

8.3 Deep Learning Methods in Relevance Model

Deep learning based methods [80, 206, 207] have achieved better performance than traditional methods in relevance-ranking. They consist of Embedding and MLP and BERT based Relevance Models.

8.3.1 Embedding and MLP-Based Relevance Model. As illustrated in Table 5, Embedding and MLP-based approaches [80, 206, 207] are prevalent in relevance-ranking. DSSM [80] adopts a two-tower architecture to learn a Deep Structured Semantic Model, where textual features are input to the query tower and item tower, and the labeled data is the click-through data. MASM [206] employs a query tower, an item tower, and an MLP to learn a relevance model leveraging click-through data in E-commerce search. Most previous work [80] learns relevance models from click-through data that are cheap and abundant. However, semantic relevance is different from CTR prediction, and click behavior may be noisy and misleading. To solve these problems, MASM employs a two-stage training process: it first pre-trains on carefully constructed samples from click-through data, then fine-tunes on high-quality human-labeled relevance data.

8.3.2 BERT Based Relevance Model. BERT based relevance models [108, 207, 214] are widely applied in industry search. ReprBERT [207] distills knowledge from an interaction-based BERT model into an efficient representation-based relevance model for E-Commerce search. SPM [214] utilizes BERT for relevance matching with a richly structured document. DeepBoW [108] is a two-tower architecture that employs a Transformer model to encode queries and products into sparse Bag-of-Words representations, after which semantic relevance scores are computed based on query and document representations.

8.4 LLM-Based Relevance Model

Recently, LLM-based approaches [125, 231] have been applied to relevance-ranking, yielding notable improvements in commercial search engines. These methods can be categorized into two types according to the modalities of features they use:

- Textual LLM-based methods. For instance, ELLM-rele [231] employs an explainable LLM for relevance modeling, which decomposes relevance learning into intermediate steps framed as Chain-of-Thought reasoning. Then it uses a distillation module to transfer the LLM's knowledge to interaction-based and representation-based student models. GenFR [209] trains an LLM-based relevance model and transfers its capabilities to BERT. This LLM-based model is deployed in the Tencent QQ Browser search engine to handle long-tail queries via query-based on-demand computing. Walmart [125] fine-tunes a billion-parameter LLM on human-labeled query-item pairs to learn the relevance model.
- MLLM-based methods. HCMRM [51] is an MLLM-based relevance model designed for search ads in short-video platforms. It enhances relevance matching between queries and video advertisements in Kuaishou.

8.5 Future Work in Relevance-Ranking

The promising research direction is: LLM-based Relevance-ranking.

- LLM-based relevance-ranking. Applying LLMs to relevance-ranking has emerged as a highly promising direction, with pioneering work [125, 231] demonstrating substantial gains over traditional methods. To translate this potential into practical deployment, critical future research directions include: (1) Exploring advanced fine-tuning strategies and novel architectures specifically tailored for the task of relevance-ranking, moving beyond generic LLM

capabilities. (2) Developing efficient inference techniques, such as model distillation and quantization, to make LLM-based relevance-ranking feasible for real-time, large-scale industrial applications [209].

9 Evaluation of Ranking Systems

This section introduces evaluation methods for industrial ranking systems. Section 9.1 provides a general overview of evaluation in industrial ranking systems: combining offline and online evaluations. Section 9.2 then describes commonly used public datasets. Section 9.3 introduces offline evaluation methods for the multiple stages of industrial ranking systems (i.e., matching, pre-ranking, fine-grained ranking, post-ranking, and relevance-ranking stages). Section 9.4 examines online evaluation methods.

9.1 Overview of Evaluation in Ranking Systems

This subsection outlines evaluation practices in industrial ranking systems.

Evaluation Paradigm in Industrial Ranking Systems. Industrial ranking systems typically integrate offline and online evaluation to assess model performance. During offline evaluation, models are assessed on offline data using predefined offline metrics (e.g., Recall@K, AUC). During online evaluation, models are tested via A/B testing using business metrics (e.g., GMV, orders, user dwell time, user retention, and ad revenue) in online services.

Correlations between Offline and Online Evaluation. The relationship and distinct roles of offline and online evaluation are summarized as follows:

- Offline evaluation serves as a proxy for online performance. Empirical studies indicate that a model excelling in well-designed offline evaluations tends to perform better in online evaluation [210, 240]. The key challenge is refining the offline evaluation pipeline, including training and testing dataset design and metric selection, to maximize consistency with online evaluation results [240].
- Offline evaluation is indispensable due to its shorter testing time and absence of negative impact on online metrics. (1) Offline evaluation can be completed quickly (e.g., within several minutes or hours). In contrast, online evaluation often requires several days or weeks to achieve statistical significance. Thus, offline evaluation is ideal for screening numerous model candidates and selecting the most promising model for online testing. (2) Offline evaluation does not affect user experience or business metrics in online services. By filtering out models likely to degrade online performance, it prevents potentially harmful experiments from reaching the production environment.
- Despite its utility, offline evaluation remains an approximation. Online A/B testing provides the definitive measure of a model's value by capturing its true impact on online service performance. Therefore, online evaluation is the essential final step and serves as the ground truth for models deployed in industrial settings.

9.2 Datasets

To ensure reproducibility, ranking research in industry often validates method effectiveness on both public and private industrial datasets [182, 217, 240]. Table 6 lists the most widely used public datasets for ranking, which are collected from online services and have been extensively used in experiments for matching [31, 98, 148, 242], Pre-ranking [236], Fine-grained ranking [29, 142, 182, 215, 240], and Post-ranking [138]. Research [98, 138, 148, 182, 217, 236, 240] has demonstrated that performance improvements on these public datasets correlate positively with gains on industrial datasets and in online testing.

Table 6. Public Datasets for Ranking

Datasets	Datasets Link	Related Work
Amazon	https://jmcauley.ucsd.edu/data/amazon/	[98, 148, 182, 215, 240]
MovieLens	https://grouplens.org/datasets/movielens/	[182, 215, 217, 240]
Taobao Rec	https://tianchi.aliyun.com/dataset/649	[29, 31, 142, 242]
Taobao Ads	https://tianchi.aliyun.com/dataset/56	[50, 217, 236, 240]
Ali-CCP	https://tianchi.aliyun.com/dataset/408	[120, 171, 179]
Criteo	http://labs.criteo.com/2014/02/kaggle-display-advertising-challenge-dataset	[24, 182, 217]
Avito	https://www.kaggle.com/c/avito-context-ad-clicks/data	[48, 132, 133, 150]
Re-ranking	https://github.com/rank2rec/rerank	[138]

9.3 Offline Evaluation

This section introduces state-of-the-art offline evaluation methods in industrial ranking systems. We describe the construction of the offline evaluation dataset and the widely used metrics for matching, Pre-ranking, Fine-grained ranking, Post-ranking, and Relevance-ranking.

9.3.1 Offline Evaluation Dataset. In offline evaluation, researchers typically sort the dataset chronologically and split it into training and testing sets based on time.

For each user request u, the ground truth set is denoted as $T = \{t_1, t_2, ..., t_M\}$. The top K results returned by the model are $R = \{r_1, r_2, ..., r_K\}$. Evaluation metrics are calculated per request, and the average metric across all requests measures the overall quality of the model.

9.3.2 Offline Evaluation Metrics for Matching. Widely used offline evaluation metrics for matching include Recall@K, HR@K, MRR@K, and NDCG@K [130, 148, 210, 212].

In matching, the ground truth set $T = \{t_1, t_2, ..., t_M\}$ comprises items that received positive user feedback (e.g., clicks), and the top K results returned by the matching model is $R = \{r_1, r_2, ..., r_K\}$.

The metrics for each user u are defined as follows:

—Recall@K [79, 105, 210]. It measures the average probability of including the ground truth item in the top-K retrieved results, as defined in Equation (8):

$$Recall@K = \frac{\sum_{i=1}^{K} I(r_i \in T)}{|T|}, \tag{8}$$

where $I(x)$ is the indicator function that returns 1 if x is true and 0 otherwise, and $|T|$ is size of the ground truth set.

—HR@K (Hit Ratio) [110, 114, 163, 212, 215]. It measures the proportion of requests where at least one of the top-K retrieved items is in the ground truth set, as given in Equation (9):

$$HR@K = I\left(\left|T \bigcap R\right| > 0\right), \tag{9}$$

where $|T \bigcap R|$ denotes the size of intersection set of T and R.

—MRR@K (Mean Reciprocal Rank) [114, 130, 212]. It is the reciprocal of the rank position of the first relevant item, as defined in Equation (10):

$$MRR@K = \frac{1}{rank_i}, \tag{10}$$

where $rank_i$ refers to the rank position of the first ground truth item in R. If the top-K retrieved results do not contain any ground truth item, $MRR@K$ is zero.

—NDCG@K (Normalized Discounted Cumulative Gain) [114, 148, 215, 238]. It measures whether the target ground truth items are sorted at the top of the retrieved results, as given in

Equation (11):

$$NDCG@K = \frac{\sum_{i=1}^{K} \frac{I(r_i \in T)}{log_2(i+1)}}{\sum_{i=1}^{M} \frac{1}{log_2(i+1)}}, \tag{11}$$

where $I(x)$ is the indicator function.

9.3.3 Offline Evaluation Metrics for Pre-Ranking. Common offline metrics for pre-ranking include AUC, **Grouped AUC (GAUC)**, HR@K, NDCG@K, MAP@K [185, 228, 236]. These can be categorized into accuracy-based and consistency-based metrics.

Accuracy-based metrics (e.g., AUC, GAUC [185, 228]) evaluate the pre-ranking model's classification ability. For Multi-objective based pre-ranking models, the label is typically user's feedback [185].

Consistency-based metrics (e.g., HR@K, NDCG@K, MAP@K [236]) measure the consistency between the pre-ranking model's output and the results from the subsequent fine-grained ranking model when the inputs are the same (i.e., candidate items for pre-ranking). Here, the ground truth item set $T = \{t_1, t_2, ..., t_M\}$ consists of the top items selected by the Fine-grained ranking model [228, 236], and $R = \{r_1, r_2, ..., r_K\}$ is the top K results returned by the pre-ranking model.

9.3.4 Offline Evaluation Metrics for Fine-Grained Ranking. The widely used offline evaluation metrics for Fine-grained ranking include AUC, GAUC, LogLoss, **Normalized Entropy (NE)**, **Relative Information Gain (RIG)**, **Predicted CTR Over Actual CTR (PCOC)**, and **Empirical Calibration Error (ECE)** [74, 158, 197, 217, 240, 243]. Among these, AUC is the most prevalent metric [60, 217, 240].

In fine-grained ranking, where the label y_i represents user feedback (e.g., click, conversion), and p_i is the model's prediction score, evaluation focuses on two aspects:

- Ranking performance. The model should generate an optimal ranking order, assigning higher prediction scores to positive examples than to negative ones.
- Calibration performance. A well-calibrated model is essential. Calibration, which is the ratio of the average predicted score to the average empirical (i.e., actual) score in the data, measures the agreement between predicted scores and actual outcomes. In tasks such as CTR prediction, this corresponds to the ratio of the total predicted clicks to the total observed clicks. Accurate calibration is critically important in industrial applications, where well-calibrated predictions (e.g., CTR, CVR) are necessary not only for correct ranking order but also for reliable score magnitudes. For example, in CPC advertising, the ranking score is computed as $CTR \cdot bid \cdot 1{,}000$, which depends directly on having accurately calibrated CTR estimates.

Metrics for Evaluating Ranking Performance. To measure ranking performance, widely used metrics include AUC, GAUC, LogLoss, NE, and RIG. AUC and GAUC are effective metrics for measuring ranking quality, though they do not account for calibration. In contrast, LogLoss, NE, and RIG can evaluate both ranking and calibration performance.

- AUC [60, 217, 240]. AUC measures a model's ability to distinguish between positive and negative classes. It represents the probability that the model will rank a randomly chosen positive instance higher than a randomly chosen negative one. A higher AUC indicates better ranking performance.
- GAUC [240, 243]. To measure the ranking performance in a subset of the dataset (i.e., group), we used the metric GAUC. GAUC aggregates data into groups (e.g., by user), calculates the AUC within each group, and computes a weighted average across groups, where weights are

typically proportional to impression or click counts. It is defined in Equation (12):

$$GAUC = \frac{\sum_{i=1}^{G} w_i \cdot AUC_i}{\sum_{i=1}^{G} w_i}, \tag{12}$$

where G is the number of groups, w_i is the weight of group i, and AUC_i is the AUC of group i. The model's ranking performance is better when GAUC is higher.

—LogLoss [217]. The log loss (Negative Log-Likelihood Loss) calculates the cross-entropy loss between predicted scores and true labels, as defined in Equation (13):

$$LogLoss = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log p_i + (1 - y_i)\log(1 - p_i)), \tag{13}$$

where $y_i \in \{0, 1\}$ is the label, $p_i \in [0, 1]$ is the predicted score by the model, and N is the number of training examples. It quantifies the penalty for prediction error. Lower LogLoss indicates better ranking performance.

—NE [74, 215]. It is the log loss normalized by the entropy of the dataset's average empirical score (e.g., average CTR). This normalization makes NE insensitive to the empirical score. It is defined as Equation (14):

$$NE = \frac{-\frac{1}{N}\sum_{i=1}^{N}(y_i \log p_i + (1 - y_i)\log(1 - p_i))}{-(p \log p + (1 - p)\log(1 - p))}, \tag{14}$$

where p is the average empirical score of the dataset. Lower NE indicates better predictive performance.

—RIG [74]. It is derived from NE, as defined in Equation (15):

$$RIG = 1 - NE. \tag{15}$$

A higher RIG corresponds to better ranking performance.

Metrics for Evaluating Calibration Performance. The widely used metrics for assessing the calibration of ranking models are PCOC and ECE.

—PCOC [158]. PCOC is the ratio of the average predicted score to the average true label, as defined in Equation (16):

$$PCOC = \frac{\frac{1}{N}\sum_{i=1}^{N} p_i}{\frac{1}{N}\sum_{i=1}^{N} y_i}. \tag{16}$$

Calibration performance improves as PCOC approaches 1.

—ECE [8, 158, 197]. ECE is a general calibration metric for both regression and classification tasks. We sort the data by prediction scores, divide them into M bins, and calculate ECE as the weighted average of the absolute difference between the average label and the average prediction per bin. It is defined in Equation (17):

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{N}\left|\frac{\sum_{i \in B_m} y_i}{|B_m|} - \frac{\sum_{i \in B_m} p_i}{|B_m|}\right|, \tag{17}$$

where N is the number of examples, and B_m is the mth bin. Calibration performance improves as ECE approaches 0.

9.3.5 Offline Evaluation Metrics for Post-Ranking. Common offline evaluation metrics for post-ranking include NDCG@K, Recall@K, Precision@K, and MAP@K [16, 138, 150].

In post-ranking, for a user u, the ground truth set $T = \{t_1, t_2, ..., t_M\}$ contains items that received positive feedback (e.g., clicks). The top-K results returned by the post-ranking model form the set $R = \{r_1, r_2, ..., r_K\}$.

—Precision@K [138]. It is defined as the fraction of positive items within the top-K results, as defined in Equation (18):

$$Precision@K = \frac{\sum_{i=1}^{K} I(r_i \in T)}{K}, \tag{18}$$

where $I(x)$ is the indicator function that returns 1 if x is true and 0 otherwise.

—MAP@K (Mean Average Precision) [138]. It is defined in Equation (19):

$$MAP@K = \frac{\sum_{i=1}^{K} Precision@i \cdot I(r_i \in T)}{K}, \tag{19}$$

where $I(x)$ is the indicator function.

9.3.6 Offline Evaluation Metrics for Relevance-Ranking. Relevance-ranking is often treated as a binary classification task, with human annotations ("Relevant," "Not Relevant") as labels. The AUC is the most widely used evaluation metric for this task [206, 207].

9.4 Online Evaluation

9.4.1 Online A/B Testing. In online A/B testing [170], users of an online service are randomly divided into two equal groups: a control group and an experimental group. The baseline model serves the control group, while the experimental model serves the experimental group concurrently. We measure the performance of the models based on online business metrics.

9.4.2 Online Evaluation Metrics in A/B Testing. The online evaluation metrics are business-oriented metrics and vary by application domain.

In E-commerce applications, key metrics include:

—Orders [21, 105]. It is the total number of orders.
—GMV [21, 105]. It is the total monetary value of all paid orders.

In Web Search applications, the most important online metrics are listed as follows:

—Clicks [115]. It is the total number of users clicks.
—Query-change rate [245]. It measures how often users modify their initial search query during a session, and a lower rate is better.

In content applications, primary metrics include:

—Active days [62, 196, 245]. It is the average number of days a user is active within the application during the experiment period, closely related to the **Daily Active Users (DAU)** metric.
—Stay time [22, 38, 62, 245]. It is the average daily time users spend in the application.
—Engagement metrics [62, 63, 196, 215, 245]. They are users' average finishing, commenting, liking, and sharing rate.

In Online advertising, crucial metrics are:

—Ad Revenue [21, 31, 240, 245]. It is the total amount advertisers pay for displayed ads.
—eCPM [240, 245]. It represents the ad revenue generated per 1,000 advertising impressions.

10 Future Research Directions and Open Issues

This section outlines prospective research directions and open issues within industrial ranking systems, including LLM for Ranking System, the Cold-start problem, the Exploration and Exploitation problem, The Feedback loop problem and Long-term engagement optimization.

10.1 LLM for Ranking System

LLMs have emerged as a prominent research focus in ranking systems. They have demonstrated initial success across matching [86, 115, 135, 148, 215], ranking [86, 193, 215, 217, 218], and relevance-ranking [125, 231] stages in industrial ranking systems, leading to significant improvements in online metrics.

Challenges of Deploying LLM in Industrial Ranking Systems. Despite their potential, the large-scale deployment of LLMs in production environments still faces significant challenges:

- —Latency. LLM-based ranking models tend to be more complex and introduce higher system latency. Substantial engineering effort is required to reduce serving latency in systems that serve billions of users.
- —Hardware costs. LLM-based models generally demand considerably more hardware resources (e.g., GPUs) [233, 235], creating a tradeoff between performance gains and infrastructure costs.

Future work on LLM-based methods must therefore focus on further improving online metrics while simultaneously reducing latency and hardware costs.

Promising research directions include: Generative Ranking, Content Understanding, Content Generation, Feature Interactions Scaling, Sequence Modeling Scaling, Conversational Ranking Systems, and Latency and Hardware Cost Optimization.

10.1.1 Generative Ranking. Generative Ranking-based methods [193, 215], which directly decode the identifiers of the target candidates in matching [6, 135, 148, 204, 215] and ranking [42, 66, 81, 193, 215] stages, have shown initial promise in ranking systems. Key open problems remain in improving generative model performance and designing effective training data.

10.1.2 Content Understanding. LLM-based models, including Textual and MLLMs, have achieved state-of-the-art performance in understanding item content (e.g., textual, image, and audio) [71, 113, 159, 220], which is critical for ranking systems. They enhance the understanding of users as follows: first, an LLM processes the user's historically interacted items to generate a list of item embeddings; then, sequential models (e.g., Transformer) extract user interests from these embeddings [71, 86]. Finally, matching scores between users and items are calculated based on the similarity of their respective embeddings from a content perspective.

- —*Textual LLMs*. Textual LLMs (e.g., GPT-3 [15], ChatGPT [131]) have achieved remarkable success in NLP. Researchers have applied them to matching [86, 115, 219], ranking [26, 86, 251], and relevance-ranking [125, 231] in industrial ranking systems, yielding significant metric improvements.
- —*MLLMs*. MLLMs [2, 10, 145, 172] scale up Transformer-based models to process diverse data types (e.g., text, image, audio, and video features), offering more powerful content understanding than textual LLMs. MLLMs have demonstrated success in matching [71, 113, 220], ranking [113, 159], and relevance-ranking [51] in industrial ranking systems.

  LLM-based methods provide valuable guidance for future industrial content-based ranking algorithms: (1) They prove the effectiveness of applying LLMs across multiple stages (e.g., matching, ranking, and relevance-ranking) in ranking systems, demonstrating strong performance on content features. (2) They offer clear technical pathways, showing that continued

pre-training and target-specific fine-tuning of pre-trained LLMs can enhance their ranking performance.

10.1.3 Content Generation. LLMs exhibit compelling abilities in generating diverse content, such as text (e.g., ChatGPT [131]), images (e.g., Stable Diffusion [153]), and videos (e.g., Sora [136]). The AI Generated Content, which may attract user interest, holds potential as candidate items for search engines, recommender systems, and online advertising. A promising direction in advertising is LLM-based creative generation [87], which aims to produce engaging ad creatives.

10.1.4 Feature Interactions Scaling. Inspired by scaling laws [91] in the LLM domain, researchers have designed larger ranking models. In feature interaction, some research work (e.g., Wukong [217], DHEN [218], Hiformer [63], RankMixer [245]) has demonstrated scaling laws in the feature interaction module, achieving promising results. Future research should seek new feature interaction model structures with superior performance and scalability.

10.1.5 Sequence Modeling Scaling. In the sequence modeling direction, some research work (LONGER [21], STCA [62]) scales up Transformers for long sequence modeling to establish a scaling law in sequence modeling. Promising directions involve discovering new sequential model structures that offer better performance and scaling capabilities.

10.1.6 Conversational Ranking System. Developing conversational recommendation and search engines [84, 167] like ChatGPT [131] is a promising direction [226]. Through interactive dialogue, such systems can better understand user needs and substantially improve the user experience.

10.1.7 Latency and Hardware Cost Optimization. LLM-based models typically incur higher system latency and require more hardware resources (e.g., GPUs). Therefore, optimizing for efficiency is a prerequisite for industrial deployment. Promising research directions to mitigate these costs include:

- Caching. A common strategy involves offline precomputation to reduce online load. For example, item embeddings generated by an LLM can be computed offline once and stored in a cache for direct retrieval during online serving [86, 115].
- On-demand computing. LLM-based models can be deployed selectively to serve user requests where they are most needed. For example, GenFR [209] uses the query-based on-demand computing, applying LLMs to process long-tail queries that benefit most from advanced semantic understanding.
- Knowledge distillation. This technique transfers the knowledge from a large teacher LLM into a smaller and faster student model [207, 209]. The distilled model retains much of the performance while being far more efficient, making it suitable for high-throughput online serving.
- Serving infrastructure optimization. Significant gains can be achieved by optimizing the inference engine. Techniques such as KV caching [141] and Flash Attention [40] are critical for reducing latency and increasing the throughput of LLM inference in production.

10.2 The Cold-Start Problem

The Cold-start problem, which aims to improve ranking for new items or new users [39, 230], remains a persistent challenge due to the constant influx of new items and users. For cold-start items, researchers typically leverage item content information (e.g., titles, images) or use the Exploration and Exploitation methods (e.g., Contextual-Bandit [104]). For the cold-start users problem, methods like POSO [39] introduce user-group-specialized sub-modules to enhance the personalization of

cold-start users within the ranking model. Developing more sophisticated methods for cold-start items and users holds significant value.

10.3 Exploration and Exploitation Problem

The industrial information systems often recommend popular items or those similar to users' historical behaviors, leading to two issues: (1) New items struggle to gain exposure; (2) users encounter a homogeneous set of items. To evaluate new items and discover user interests, researchers employ the Exploration and Exploitation paradigm (e.g., Multi-armed Bandit [123], Contextual-Bandit [104]) to collect real-time user feedback on new items via a small amount of exploration traffic. Based on the users' response to randomly selected items on a small amount of exploration traffic, new popular items and users' interest can be identified and exploited on the remaining traffic. Balancing Exploration and Exploitation remains an open and critical problem in industrial ranking systems.

10.4 The Feedback Loop Problem

Existing approaches often train the ranking models based on user feedback on the items selected by the previous ranking model. The biased exposure of items leads to the Feedback loop problem in industrial ranking systems: popular items become increasingly dominant, while long-tail items receive even less exposure [58, 118, 228]. This self-reinforcing cycle causes ranking models to generate progressively more biased results over time. Addressing it is therefore paramount. To break this loop, recent methods leverage alternative data sources. ZEUS [58] employs self-supervised learning on users' spontaneous behaviors to break the feedback loop problem and generates a pre-trained model, and then fine-tunes the pre-trained model based on users' implicit feedback in recommendations for ranking. EntireSpace [118] incorporates purchase data from other scenarios (e.g., search) to learn the ranking model in the recommendation scenario, and rebuilds the fine-grained ranking training data on the entire data space with samples from previous stages of the ranking system. Novel approaches to mitigate this loop are of significant importance.

10.5 Long-Term Reward Optimization Problem

The information system aims to optimize users' long-term rewards (e.g., Average Active Days [196], DAU) within an application. However, most ranking methods are designed to optimize instant user feedback (e.g., clicks or orders). Some research [28, 83, 248] has employed DRL to optimize long-term rewards in ranking systems. For example, FeedRec [248] introduces an RL framework to improve long-term user engagement metrics (e.g., average clicks or browse depth per session, average return time). Future work exploring practical methods for long-term reward optimization remains a promising research direction.

11 Conclusions

This article provides a systematic review of Deep learning based models widely used in industrial Search engines, Recommender systems, and Online advertising ranking systems. Furthermore, we introduce new perspectives such as the application of LLMs for industrial ranking. We also discuss future research directions and open issues in industrial ranking systems. We hope this article offers readers in both research and industry a comprehensive understanding of key problems and state-of-the-art approaches in industrial ranking systems, while also shedding light on promising directions for future research.

Acknowledgements

We thank the supercomputing system in the Supercomputing Center of Wuhan University for its support of numerical calculations.

References

[1] Mustafa Abdool, Malay Haldar, Prashant Ramanathan, Tyler Sax, Lanbo Zhang, Aamir Manaswala, Lynn Yang, Bradley Turnbull, Qing Zhang, and Thomas Legrand. 2020. Managing diversity in Airbnb search. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2952–2960.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. arXiv:2303.08774. Retrieved from https://arxiv.org/abs/2303.08774

[3] Qingyao Ai, Keping Bi, Jiafeng Guo, and W. Bruce Croft. 2018. Learning a deep listwise context model for ranking refinement. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, 135–144.

[4] Rohan Anil, Sandra Gadanho, Da Huang, Nijith Jacob, Zhuoshu Li, Dong Lin, Todd Phillips, Cristina Pop, Kevin Regan, Gil I. Shamir, et al. 2022. On the factory floor: ML engineering for industrial-scale ads recommendation models. arXiv:2209.05310. Retrieved from https://arxiv.org/abs/2209.05310

[5] Ashwinkumar Badanidiyuru, Andrew Evdokimov, Vinodh Krishnan, Pan Li, Wynn Vonnegut, and Jayden Wang. 2021. Handling many conversions per click in modeling delayed feedback. arXiv:2101.02284. Retrieved from https://arxiv.org/abs/2101.02284

[6] Anirudhan Badrinath, Prabhat Agarwal, Laksh Bhasin, Jaewon Yang, Jiajing Xu, and Charles Rosenberg. 2025. PinRec: Outcome-conditioned, multi-token generative retrieval for industry-scale recommendation systems. arXiv:2504.10507. Retrieved from https://arxiv.org/abs/2504.10507

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. arXiv:1409.0473. Retrieved from https://arxiv.org/abs/1409.0473

[8] Aijun Bai, Rolf Jagerman, Zhen Qin, Le Yan, Pratyush Kar, Bing-Rong Lin, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2023. Regression compatible listwise objectives for calibrated ranking with binary relevance. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 4502–4508.

[9] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. arXiv:2309.16609. Retrieved from https://arxiv.org/abs/2309.16609

[10] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. 2025. Qwen2.5-vl technical report. arXiv:2502.13923. Retrieved from https://arxiv.org/abs/2502.13923

[11] Oren Barkan and Noam Koenigstein. 2016. Item2vec: Neural item embedding for collaborative filtering. In *Proceedings of the 2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 1–6.

[12] Irwan Bello, Sayali Kulkarni, Sagar Jain, Craig Boutilier, Ed Chi, Elad Eban, Xiyang Luo, Alan Mackey, and Ofer Meshi. 2018. Seq2slate: Re-ranking and slate optimization with RNNs. arXiv:1810.02019. Retrieved from https://arxiv.org/abs/1810.02019

[13] Weijie Bian, Kailun Wu, Lejian Ren, Qi Pi, Yujing Zhang, Can Xiao, Xiang-Rong Sheng, Yong-Nan Zhu, Zhangming Chan, Na Mou, et al. 2022. CAN: Feature co-action network for click-through rate prediction. In *Proceedings of the 15th ACM International Conference on Web Search and Data Mining*, 57–65.

[14] Xingyan Bin, Jianfei Cui, Wujie Yan, Zhichen Zhao, Xintian Han, Chongyang Yan, Feng Zhang, Xun Zhou, Wu Qi, and Zuotao Liu. 2025. Real-time indexing for large-scale recommendation by streaming vector quantization retriever. arXiv:2501.08695. Retrieved from https://arxiv.org/abs/2501.08695

[15] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing System (NIPS '20)*, 1877–1901.

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

[17] Yue Cao, Xiaojiang Zhou, Jiaqi Feng, Peihao Huang, Yao Xiao, Dayao Chen, and Sheng Chen. 2022. Sampling is all you need on modeling long-term user behaviors for CTR prediction. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, 2974–2983.

[18] Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 335–336.

[19] Yukuo Cen, Jianwei Zhang, Xu Zou, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. Controllable multi-interest framework for recommendation. arXiv:2005.09347. Retrieved from https://arxiv.org/abs/2005.09347

[20] Yukuo Cen, Xu Zou, Jianwei Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Representation learning for attributed multiplex heterogeneous network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1358–1368.

[21] Zheng Chai, Qin Ren, Xijun Xiao, Huizhi Yang, Bo Han, Sijun Zhang, Di Chen, Hui Lu, Wenlin Zhao, Lele Yu, et al. 2025. LONGER: Scaling up long sequence modeling in industrial recommenders. arXiv:2505.04421. Retrieved from https://arxiv.org/abs/2505.04421

[22] Jianxin Chang, Chenbin Zhang, Zhiyi Fu, Xiaoxue Zang, Lin Guan, Jing Lu, Yiqun Hui, Dewei Leng, Yanan Niu, Yang Song, et al. 2023. TWIN: TWo-stage interest network for lifelong user behavior modeling in CTR prediction at Kuaishou. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 3785–3794.

[23] Jianxin Chang, Chenbin Zhang, Yiqun Hui, Dewei Leng, Yanan Niu, Yang Song, and Kun Gai. 2023. PEPnet: Parameter and embedding personalized network for infusing with personalized prior information. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 3795–3804.

[24] Olivier Chapelle. 2014. Modeling delayed feedback in display advertising. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1097–1105.

[25] Ben Chen, Xian Guo, Siyuan Wang, Zihan Liang, Yue Lv, Yufei Ma, Xinlong Xiao, Bowen Xue, Xuxin Zhang, Ying Yang, et al. 2025. OneSearch: A preliminary exploration of the unified end-to-end generative framework for e-commerce search. arXiv:2509.03236. Retrieved from https://arxiv.org/abs/2509.03236

[26] Junyi Chen, Lu Chi, Bingyue Peng, and Zehuan Yuan. 2024. HLLM: Enhancing sequential recommendations via hierarchical large language models for item and user modeling. arXiv:2409.12740. Retrieved from https://arxiv.org/abs/2409.12740

[27] Laming Chen, Guoxin Zhang, and Eric Zhou. 2018. Fast greedy map inference for determinantal point process to improve recommendation diversity. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS '18)*, 5622–5633.

[28] Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed. H. Chi. 2019. Top-K off-policy correction for a REINFORCE recommender system. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*, 456–464.

[29] Qiwei Chen, Yue Xu, Changhua Pei, Shanshan Lv, Tao Zhuang, and Junfeng Ge. 2022. Efficient long sequential user data modeling for click-through rate prediction. arXiv:2209.12212. Retrieved from https://arxiv.org/abs/2209.12212

[30] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior sequence transformer for e-commerce recommendation in Alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*, 1–4.

[31] Rihan Chen, Bin Liu, Han Zhu, Yaoxuan Wang, Qi Li, Buting Ma, Qingbo Hua, Jun Jiang, Yunlong Xu, Hongbo Deng, et al. 2022. Approximate nearest neighbor search under neural similarity metric for large-scale recommendation. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, 3013–3022.

[32] Weijian Chen, Yulong Gu, Zhaochun Ren, Xiangnan He, Hongtao Xie, Tong Guo, Dawei Yin, and Yongdong Zhang. 2019. Semi-supervised user profiling with heterogeneous graph attention networks. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI '19)*, 2116–2122.

[33] Xiangyi Chen, Kousik Rajesh, Matthew Lawhon, Zelun Wang, Hanyu Li, Haomiao Li, Saurabh Vishwas Joshi, Pong Eksombatchai, Jaewon Yang, Yi-Ping Hsu, et al. 2025. PinFM: Foundation model for user activity sequences at a billion-scale visual discovery platform. arXiv:2507.12704. Retrieved from https://arxiv.org/abs/2507.12704

[34] Xiaocong Chen, Lina Yao, Julian McAuley, Guanglin Zhou, and Xianzhi Wang. 2023. Deep reinforcement learning in recommender systems: A survey and new perspectives. *Knowledge-Based Systems* 264 (2023), 110335.

[35] Yu Chen, Jiaqi Jin, Hui Zhao, Pengjie Wang, Guojun Liu, Jian Xu, and Bo Zheng. 2022. Asymptotically unbiased estimation for delayed feedback modeling via label correction. In *Proceedings of the ACM Web Conference 2022*, 369–379.

[36] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 7–10.

[37] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555. Retrieved from https://arxiv.org/abs/1412.3555

[38] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for YouTube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, 191–198.

[39] Shangfeng Dai, Haobin Lin, Zhichen Zhao, Jianying Lin, Honghuan Wu, Zhe Wang, Sen Yang, and Ji Liu. 2021. POSO: Personalized cold start modules for large-scale recommender systems. arXiv:2108.04690. Retrieved from https://arxiv.org/abs/2108.04690

[40] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS '22)*, 16344–16359.

[41] Romain Deffayet, Thibaut Thonet, Jean-Michel Renders, and Maarten De Rijke. 2023. Generative slate recommendation with reinforcement learning. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, 580–588.

[42] Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. 2025. OneRec: Unifying retrieve and rank with generative recommender and iterative preference alignment. arXiv:2502.18965. Retrieved from https://arxiv.org/abs/2502.18965

[43] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 4171–4186.

[44] Miao Fan, Jiacheng Guo, Shuai Zhu, Shuo Miao, Mingming Sun, and Ping Li. 2019. MOBIUS: Towards the next generation of Query-Ad matching in Baidu's sponsored search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2509–2517.

[45] Shaohua Fan, Junxiong Zhu, Xiaotian Han, Chuan Shi, Linmei Hu, Biyu Ma, and Yongliang Li. 2019. Metapath-guided heterogeneous graph neural network for intent recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2478–2486.

[46] Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting LLMs: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 6491–6501.

[47] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph neural networks for social recommendation. In *Proceedings of the World Wide Web Conference*, 417–426.

[48] Zhifang Fan, Dan Ou, Yulong Gu, Bairan Fu, Xiang Li, Wentian Bao, Xin-Yu Dai, Xiaoyi Zeng, Tao Zhuang, and Qingwen Liu. 2022. Modeling users' contextualized page-wise feedback for click-through rate prediction in e-commerce search. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, 262–270.

[49] Yufei Feng, Yu Gong, Fei Sun, Junfeng Ge, and Wenwu Ou. 2021. Revisit recommender system in the permutation prospective. arXiv:2102.12057. Retrieved from https://arxiv.org/abs/2102.12057

[50] Yufei Feng, Fuyu Lv, Weichen Shen, Menghan Wang, Fei Sun, Yu Zhu, and Keping Yang. 2019. Deep session interest network for click-through rate prediction. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 2301–2307.

[51] Guobing Gan, Kaiming Gao, Li Wang, Shen Jiang, and Peng Jiang. 2025. HCMRM: A high-consistency multimodal relevance model for search ads. In *Companion Proceedings of the ACM on Web Conference 2025*, 201–210.

[52] Weihao Gao, Xiangjun Fan, Chong Wang, Jiankai Sun, Kai Jia, Wenzhi Xiao, Ruofan Ding, Xingyan Bin, Hui Yang, and Xiaobing Liu. 2020. Deep retrieval: Learning a retrievable structure for large-scale recommendations. arXiv:2007.07203. Retrieved from https://arxiv.org/abs/2007.07203

[53] Tiezheng Ge, Liqin Zhao, Guorui Zhou, Keyu Chen, Shuying Liu, Huimin Yi, Zelin Hu, Bochao Liu, Peng Sun, Haoyu Liu, et al. 2018. Image matters: Visually modeling user behaviors using advanced model server. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2087–2095.

[54] Mihajlo Grbovic and Haibin Cheng. 2018. Real-time personalization using embeddings for search ranking at Airbnb. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 311–320.

[55] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 855–864.

[56] Siyu Gu, Xiang-Rong Sheng, Ying Fan, Guorui Zhou, and Xiaoqiang Zhu. 2021. Real negatives matter: Continuous training with real negatives for delayed feedback modeling. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2890–2898.

[57] Yulong Gu. 2021. Attentive neural point processes for event forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence* 35 (2021), 7592–7600.

[58] Yulong Gu, Wentian Bao, Dan Ou, Xiang Li, Baoliang Cui, Biyu Ma, Haikuan Huang, Qingwen Liu, and Xiaoyi Zeng. 2021. Self-supervised learning on users' spontaneous behaviors for multi-scenario ranking in e-commerce. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, 3828–3837.

[59] Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, and Dawei Yin. 2020. Hierarchical user profiling for e-commerce recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, 223–231.

[60] Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, Lixin Zou, Yiding Liu, and Dawei Yin. 2020. Deep multifaceted transformers for multi-objective ranking in large-scale e-commerce recommender systems. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, 2493–2500.

[61] Yulong Gu, Jiaxing Song, Weidong Liu, and Lixin Zou. 2016. HLGPS: A home location global positioning system in location-based social networks. In *Proceedings of the 2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 901–906.

[62] Lin Guan, Jia-Qi Yang, Zhishan Zhao, Beichuan Zhang, Bo Sun, Xuanyuan Luo, Jinan Ni, Xiaowen Li, Yuhang Qi, Zhifang Fan, et al. 2025. Make it long, keep it fast: End-to-end 10k-sequence modeling at billion scale on Douyin. arXiv:2511.06077. Retrieved from https://arxiv.org/abs/2511.06077

[63] Huan Gui, Ruoxi Wang, Ke Yin, Long Jin, Maciej Kula, Taibai Xu, Lichan Hong, and Ed. H. Chi. 2023. Hiformer: Heterogeneous feature interactions learning with transformers for recommender systems. arXiv:2311.05884. Retrieved from https://arxiv.org/abs/2311.05884

[64] 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 26th International Joint Conference on Artificial Intelligence*, 1725–1731.

[65] Tong Guo, Xuanping Li, Haitao Yang, Xiao Liang, Yong Yuan, Jingyou Hou, Bingqing Ke, Chao Zhang, Junlin He, Shunyu Zhang, et al. 2023. Query-dominant user interest network for Large-Scale search ranking. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 629–638.

[66] Xian Guo, Ben Chen, Siyuan Wang, Ying Yang, Chenyi Lei, Yuqing Ding, and Han Li. 2025. OneSug: The unified end-to-end generative framework for e-commerce query suggestion. arXiv:2506.06913. Retrieved from https://arxiv.org/abs/2506.06913

[67] Malay Haldar, Mustafa Abdool, Prashant Ramanathan, Tao Xu, Shulin Yang, Huizhong Duan, Qing Zhang, Nick Barrow-Williams, Bradley C. Turnbull, Brendan M. Collins, et al. 2019. Applying deep learning to Airbnb search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1927–1935.

[68] Malay Haldar, Prashant Ramanathan, Tyler Sax, Mustafa Abdool, Lanbo Zhang, Aamir Mansawala, Shulin Yang, Bradley Turnbull, and Junshuo Liao. 2020. Improving deep learning for Airbnb search. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2822–2830.

[69] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 1024–1034.

[70] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

[71] Ruining He, Lukasz Heldt, Lichan Hong, Raghunandan Keshavan, Shifan Mao, Nikhil Mehta, Zhengyang Su, Alicia Tsai, Yueqi Wang, Shao-Chuan Wang, et al. 2025. PLUM: Adapting pre-trained language models for industrial-scale generative recommendations. arXiv:2510.07784. Retrieved from https://arxiv.org/abs/2510.07784

[72] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 639–648.

[73] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, 173–182.

[74] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. 2014. Practical lessons from predicting clicks on ads at Facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*, 1–9.

[75] Yunlong He, Jiliang Tang, Hua Ouyang, Changsung Kang, Dawei Yin, and Yi Chang. 2016. Learning to rewrite queries. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 1443–1452.

[76] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. arXiv:1511.06939. Retrieved from https://arxiv.org/abs/1511.06939

[77] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[78] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. LoRA: Low-rank adaptation of large language models. In *Proceedings of the International Conference on Learning Representations*. Retrieved from https://openreview.net/pdf?id=nZeVKeeFYf9

[79] Jui-Ting Huang, Ashish Sharma, Shuying Sun, Li Xia, David Zhang, Philip Pronin, Janani Padmanabhan, Giuseppe Ottaviano, and Linjun Yang. 2020. Embedding-based retrieval in Facebook search. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2553–2561.

[80] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management*, 2333–2338.

[81] Yanhua Huang, Yuqi Chen, Xiong Cao, Rui Yang, Mingliang Qi, Yinghao Zhu, Qingchang Han, Yaowei Liu, Zhaoyu Liu, Xuefeng Yao, et al. 2025. Towards large-scale generative ranking. arXiv:2505.04180. Retrieved from https://arxiv.org/abs/2505.04180

[82] Yunwen Huang, Shiyong Hong, Xijun Xiao, Jinqiu Jin, Xuanyuan Luo, Zhe Wang, Zheng Chai, Shikang Wu, Yuchao Zheng, and Jingjian Lin. 2026. HyFormer: Revisiting the roles of sequence modeling and feature interaction in CTR prediction. arXiv:2601.12681. Retrieved from https://arxiv.org/abs/2601.12681

[83] Eugene Ie, Vihan Jain, Jing Wang, Sanmit Narvekar, Ritesh Agarwal, Rui Wu, Heng-Tze Cheng, Tushar Chandra, and Craig Boutilier. 2019. SLATEQ: A tractable decomposition for reinforcement learning with recommendation sets. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 2592–2599.

[84] Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. 2021. A survey on conversational recommender systems. *ACM Computing Surveys* 54, 5 (2021), 1–36.
[85] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2010. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33, 1 (2010), 117–128.
[86] Jian Jia, Yipei Wang, Yan Li, Honggang Chen, Xuehan Bai, Zhaocheng Liu, Jian Liang, Quan Chen, Han Li, Peng Jiang, et al. 2025. LEARN: Knowledge adaptation from large language model to recommendation for practical industrial application. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence and 37th Conference on Innovative Applications of Artificial Intelligence and 15th Symposium on Educational Advances in Artificial Intelligence (AAAI '25/IAAI '25/EAAI '25)*, 11861–11869.
[87] Zhipeng Jin, Wen Tao, Yafei Li, Yi Yang, Cong Han, Shuanglong Li, and Lin Liu. 2025. Large vison-language foundation model in Baidu AIGC image advertising. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2303–2312.
[88] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data* 7, 3 (2019), 535–547.
[89] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware factorization machines for CTR prediction. In *Proceedings of the 10th ACM Conference on Recommender Systems*, 43–50.
[90] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. 2021. Highly accurate protein structure prediction with AlphaFold. *Nature* 596, 7873 (2021), 583–589.
[91] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. arXiv:2001.08361. Retrieved from https://arxiv.org/abs/2001.08361
[92] Thomas N. Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. arXiv:1609.02907. Retrieved from https://arxiv.org/abs/1609.02907
[93] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
[94] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *Proceedings of the 26th International Conference on Neural Information Processing Systems—Volume 1 (NIPS '12)*, 1097–1105.
[95] Sofia Ira Ktena, Alykhan Tejani, Lucas Theis, Pranay Kumar Myana, Deepak Dilipkumar, Ferenc Huszár, Steven Yoo, and Wenzhe Shi. 2019. Addressing delayed feedback for continuous training with neural networks in CTR prediction. In *Proceedings of the 13th ACM Conference on Recommender Systems*, 187–195.
[96] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.
[97] Caiwen Li, Iskandar Ishak, Hamidah Ibrahim, Maslina Zolkepli, Fatimah Sidi, and Caili Li. 2023. Deep learning-based recommendation system: Systematic review and classification. *IEEE Access* 11 (2023), 113790–113835.
[98] Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Huan Zhao, Pipei Huang, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. 2019. Multi-interest network with dynamic routing for recommendation at Tmall. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2615–2623.
[99] Cheng Li, Yue Lu, Qiaozhu Mei, Dong Wang, and Sandeep Pandey. 2015. Click-through prediction for advertising in Twitter timeline. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1959–1968.
[100] Danwei Li, Zhengyu Zhang, Siyang Yuan, Mingze Gao, Weilin Zhang, Chaofei Yang, Xi Liu, and Jiyan Yang. 2023. AdaTT: Adaptive task-to-task fusion network for multitask learning in recommendations. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 4370–4379.
[101] Houyi Li, Zhihong Chen, Chenliang Li, Rong Xiao, Hongbo Deng, Peng Zhang, Yongchao Liu, and Haihong Tang. 2021. Path-based deep network for candidate item matching in recommenders. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1493–1502.
[102] Hang Li and Zhengdong Lu. 2016. Deep learning for information retrieval. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1203–1206.
[103] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 1419–1428.
[104] Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, 661–670.
[105] Sen Li, Fuyu Lv, Taiwei Jin, Guli Lin, Keping Yang, Xiaoyi Zeng, Xiao-Ming Wu, and Qianli Ma. 2021. Embedding-based product retrieval in Taobao search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 3181–3189.

[106] Xiaoxi Li, Jiajie Jin, Yujia Zhou, Yuyao Zhang, Peitian Zhang, Yutao Zhu, and Zhicheng Dou. 2025. From matching to generation: A survey on generative information retrieval. *ACM Transactions on Information Systems* 43, 3 (2025), 1–62.

[107] Zhutian Lin, Junwei Pan, Shangyu Zhang, Ximei Wang, Xi Xiao, Shudong Huang, Lei Xiao, and Jie Jiang. 2024. Understanding the ranking loss for recommendation with sparse user feedback. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 5409–5418.

[108] Zhe Lin, Jiwei Tan, Dan Ou, Xi Chen, Shaowei Yao, and Bo Zheng. 2024. Deep bag-of-words model: An efficient and interpretable relevance architecture for Chinese e-commerce. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 5398–5408.

[109] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing* 7, 1 (2003), 76–80.

[110] Chi Liu, Jiangxia Cao, Rui Huang, Kai Zheng, Qiang Luo, Kun Gai, and Guorui Zhou. 2024. KuaiFormer: Transformer-based retrieval at Kuaishou. arXiv:2411.10057. Retrieved from https://arxiv.org/abs/2411.10057

[111] Chang Liu, Qiwei Wang, Wenqing Lin, Yue Ding, and Hongtao Lu. 2024. Beyond binary preference: Leveraging bayesian approaches for joint optimization of ranking and calibration. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 5442–5453.

[112] Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval* 3, 3 (2009), 225–331.

[113] Yueyang Liu, Jiangxia Cao, Shen Wang, Shuang Wen, Xiang Chen, Xiangyu Wu, Shuang Yang, Zhaojie Liu, Kun Gai, and Guorui Zhou. 2025. LLM-alignment live-streaming recommendation. arXiv:2504.05217. Retrieved from https://arxiv.org/abs/2504.05217

[114] Yiding Liu, Yulong Gu, Zhuoye Ding, Junchao Gao, Ziyi Guo, Yongjun Bao, and Weipeng Yan. 2020. Decoupled graph convolution network for inferring substitutable and complementary items. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, 2621–2628.

[115] 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 and Data Mining*, 3365–3375.

[116] Wenhao Lu, Jian Jiao, and Ruofei Zhang. 2020. TwinBERT: Distilling knowledge to twin-structured BERT models for efficient retrieval. arXiv:2002.06275. Retrieved from https://arxiv.org/abs/2002.06275

[117] Fuyu Lv, Taiwei Jin, Changlong Yu, Fei Sun, Quan Lin, Keping Yang, and Wilfred Ng. 2019. SDM: Sequential deep matching model for online large-scale recommender system. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2635–2643.

[118] Shanshan Lyu, Qiwei Chen, Tao Zhuang, and Junfeng Ge. 2023. Entire space learning framework: Unbias conversion rate prediction in full stages of recommender system. arXiv:2303.00276. Retrieved from https://arxiv.org/abs/2303.00276

[119] 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 and Data Mining*, 1930–1939.

[120] 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 and Development in Information Retrieval*, 1137–1140.

[121] Yu. A. Malkov and Dmitry A. Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 4 (2018), 824–836.

[122] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794.

[123] James McInerney, Benjamin Lacker, Samantha Hansen, Karl Higley, Hugues Bouchard, Alois Gruson, and Rishabh Mehrotra. 2018. Explore, exploit, and explain: Personalizing explainable recommendations with bandits. In *Proceedings of the 12th ACM Conference on Recommender Systems*, 31–39.

[124] H. Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: A view from the trenches. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1222–1230.

[125] Navid Mehrdad, Hrushikesh Mohapatra, Mossaab Bagdouri, Prijith Chandran, Alessandro Magnani, Xunfan Cai, Ajit Puthenputhussery, Sachin Yadav, Tony Lee, ChengXiang Zhai, et al. 2024. Large language models for relevance judgment in product search. arXiv:2406.00247. Retrieved from https://arxiv.org/abs/2406.00247

[126] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 27th International Conference on Neural Information Processing Systems—Volume 2 (NIPS '13)*, 3111–3119.

[127] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *Proceedings of the International Conference on Machine Learning*. PMLR, 1928–1937.

[128] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing Atari with deep reinforcement learning. arXiv:1312.5602. Retrieved from https://arxiv.org/abs/1312.5602

[129] Maxim Naumov, Dheevatsa Mudigere, Hao-Jun Michael Shi, Jianyu Huang, Narayanan Sundaraman, Jongsoo Park, Xiaodong Wang, Udit Gupta, Carole-Jean Wu, Alisson G. Azzolini, et al. 2019. Deep learning recommendation model for personalization and recommendation systems. arXiv:1906.00091. Retrieved from https://arxiv.org/abs/1906.00091

[130] Priyanka Nigam, Yiwei Song, Vijai Mohan, Vihan Lakshman, Weitian Ding, Ankit Shingavi, Choon Hui Teo, Hao Gu, and Bing Yin. 2019. Semantic product search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2876–2885.

[131] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS '22)*, 27730–27744.

[132] Wentao Ouyang, Xiuwu Zhang, Li Li, Heng Zou, Xin Xing, Zhaojie Liu, and Yanlong Du. 2019. Deep spatio-temporal neural networks for click-through rate prediction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2078–2086.

[133] Wentao Ouyang, Xiuwu Zhang, Shukui Ren, Chao Qi, Zhaojie Liu, and Yanlong Du. 2019. Representation learning-assisted click-through rate prediction. arXiv:1906.04365. Retrieved from https://arxiv.org/abs/1906.04365

[134] Junwei Pan, Wei Xue, Ximei Wang, Haibin Yu, Xun Liu, Shijie Quan, Xueming Qiu, Dapeng Liu, Lei Xiao, and Jie Jiang. 2024. Ads recommendation in a collapsed and entangled world. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 5566–5577.

[135] Ming Pang, Chunyuan Yuan, Xiaoyu He, Zheng Fang, Donghao Xie, Fanyi Qu, Xue Jiang, Changping Peng, Zhangang Lin, Zheng Luo, et al. 2025. Generative retrieval and alignment model: A new paradigm for e-commerce retrieval. arXiv:2504.01403. Retrieved from https://arxiv.org/abs/2504.01403

[136] William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 4195–4205.

[137] Changhua Pei, Xinru Yang, Qing Cui, Xiao Lin, Fei Sun, Peng Jiang, Wenwu Ou, and Yongfeng Zhang. 2019. Value-aware recommendation based on reinforcement profit maximization. In *Proceedings of the World Wide Web Conference*, 3123–3129.

[138] Changhua Pei, Yi Zhang, Yongfeng Zhang, Fei Sun, Xiao Lin, Hanxiao Sun, Jian Wu, Peng Jiang, Junfeng Ge, Wenwu Ou, et al. 2019. Personalized re-ranking for recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*, 3–11.

[139] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 701–710.

[140] Qi Pi, Weijie Bian, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Practice on long sequential user behavior modeling for click-through rate prediction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2671–2679.

[141] Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. 2023. Efficiently scaling transformer inference. *Proceedings of the 6th MLSys Conference*, 606–624.

[142] Qi Pi, Xiaoqiang Zhu, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, and Kun Gai. 2020. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. arXiv:2006.05639. Retrieved from https://arxiv.org/abs/2006.05639

[143] Zhen Qin, Yicheng Cheng, Zhe Zhao, Zhe Chen, Donald Metzler, and Jingzheng Qin. 2020. Multitask mixture of sequential experts for user activity streams. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 3083–3091.

[144] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network embedding as matrix factorization: Unifying DeepWalk, LINE, PTE, and node2vec. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, 459–467.

[145] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning*. PMLR, 8748–8763.

[146] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Retrieved from https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

[147] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1, 8 (2019), 9.

[148] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, et al. 2023. Recommender systems with generative retrieval. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS '23)*, 10299–10315.

[149] Pengjie Ren, Zhumin Chen, Jing Li, Zhaochun Ren, Jun Ma, and Maarten De Rijke. 2019. RepeatNet: A repeat aware neural recommendation machine for session-based recommendation. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence and 31st Innovative Applications of Artificial Intelligence Conference and 9th AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI '19/IAAI '19/EAAI '19)*, 4806–4813.

[150] Yuxin Ren, Qiya Yang, Yichun Wu, Wei Xu, Yalong Wang, and Zhiqiang Zhang. 2024. Non-autoregressive generative models for reranking recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 5625–5634.

[151] Steffen Rendle. 2010. Factorization machines. In *Proceedings of the 2010 IEEE International Conference on Data Mining*. IEEE, 995–1000.

[152] Leonardo F. R. Ribeiro, Pedro H. P. Saverese, and Daniel R. Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 385–394.

[153] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10684–10695.

[154] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323, 6088 (1986), 533–536.

[155] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, 285–295.

[156] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. arXiv:1707.06347. Retrieved from https://arxiv.org/abs/1707.06347

[157] Ying Shan, T. Ryan Hoens, Jian Jiao, Haijing Wang, Dong Yu, and J. C. Mao. 2016. Deep crossing: Web-scale modeling without manually crafted combinatorial features. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 255–262.

[158] Xiang-Rong Sheng, Jingyue Gao, Yueyao Cheng, Siran Yang, Shuguang Han, Hongbo Deng, Yuning Jiang, Jian Xu, and Bo Zheng. 2023. Joint optimization of ranking and calibration with contextualized hybrid model. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 4813–4822.

[159] Xiang-Rong Sheng, Feifan Yang, Litong Gong, Biao Wang, Zhangming Chan, Yujing Zhang, Yueyao Cheng, Yong-Nan Zhu, Tiezheng Ge, Han Zhu, et al. 2024. Enhancing taobao display advertising with multimodal representations: Challenges, approaches and insights. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 4858–4865.

[160] 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 and Knowledge Management*, 4104–4113.

[161] Xiaowen Shi, Fan Yang, Ze Wang, Xiaoxu Wu, Muzhi Guan, Guogang Liao, Wang Yongkang, Xingxing Wang, and Dong Wang. 2023. PIER: Permutation-level interest-based end-to-end re-ranking framework in e-commerce. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 4823–4831.

[162] Zihua Si, Lin Guan, ZhongXiang Sun, Xiaoxue Zang, Jing Lu, Yiqun Hui, Xingchao Cao, Zeyu Yang, Yichen Zheng, Dewei Leng, et al. 2024. Twin v2: Scaling ultra-long user behavior sequence modeling for enhanced ctr prediction at kuaishou. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 4890–4897.

[163] Thiago Silveira, Min Zhang, Xiao Lin, Yiqun Liu, and Shaoping Ma. 2019. How good your recommender system is? A survey on evaluations in recommendation. *International Journal of Machine Learning and Cybernetics* 10, 5 (2019), 813–831.

[164] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *Nature* 550, 7676 (2017), 354–359.

[165] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 1161–1170.

[166] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 1441–1450.

[167] Yueming Sun and Yi Zhang. 2018. Conversational recommender system. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, 235–244.

[168] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems—Volume 2 (NIPS '14)*, 3104–3112.

[169] Pawel Swietojanski, Jinyu Li, and Steve Renals. 2016. Learning hidden unit contributions for unsupervised acoustic model adaptation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24, 8 (2016), 1450–1463.

[170] Diane Tang, Ashish Agarwal, Deirdre O'Brien, and Mike Meyer. 2010. Overlapping experiment infrastructure: More, better, faster experimentation. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 17–26.

[171] 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.

[172] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: A family of highly capable multimodal models. arXiv:2312.11805. Retrieved from https://arxiv.org/abs/2312.11805

[173] Ilya O. Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. 2021. MLP-mixer: An all-MLP architecture for vision. In *Proceedings of the 35th International Conference on Neural Information Processing Systems (NIPS '21)*, 24261–24272.

[174] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. arXiv:2302.13971. Retrieved from https://arxiv.org/abs/2302.13971

[175] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS '17)*, 6000–6010.

[176] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. arXiv:1710.10903. Retrieved from https://arxiv.org/abs/1710.10903

[177] Fangye Wang, Hansu Gu, Dongsheng Li, Tun Lu, Peng Zhang, and Ning Gu. 2023. Towards deeper, lighter and interpretable cross network for CTR prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2523–2533.

[178] Guoquan Wang, Qiang Luo, Weisong Hu, Pengfei Yao, Wencong Zeng, Guorui Zhou, and Kun Gai. 2025. FIM: Frequency-aware multi-view interest modeling for local-life service recommendation. arXiv:2504.17814. Retrieved from https://arxiv.org/abs/2504.17814

[179] Hao Wang, Tai-Wei Chang, Tianqiao Liu, Jianmin Huang, Zhichao Chen, Chao Yu, Ruopeng Li, and Wei Chu. 2022. ESCM2: Entire space counterfactual multi-task model for post-click conversion rate estimation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 363–372.

[180] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. Billion-scale commodity embedding for e-commerce recommendation in alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 839–848.

[181] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep and cross network for ad click predictions. In *Proceedings of the the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17)*, 1–7.

[182] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. 2021. DCN V2: Improved deep and cross network and practical lessons for web-scale learning to rank systems. In *Proceedings of the Web Conference 2021*, 1785–1797.

[183] Xu Wang, Jiangxia Cao, Zhiyi Fu, Kun Gai, and Guorui Zhou. 2024. HoME: Hierarchy of multi-gate experts for multi-task learning at Kuaishou. arXiv:2408.05430. Retrieved from https://arxiv.org/abs/2408.05430

[184] Yanshi Wang, Jie Zhang, Qing Da, and Anxiang Zeng. 2020. Delayed feedback modeling for the entire space conversion rate prediction. arXiv:2011.11826. Retrieved from https://arxiv.org/abs/2011.11826

[185] Zhe Wang, Liqin Zhao, Biye Jiang, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. 2020. Cold: Towards the next generation of pre-ranking system. arXiv:2007.16122. Retrieved from https://arxiv.org/abs/2007.16122

[186] Zhipeng Wei, Kuo Cai, Junda She, Jie Chen, Minghao Chen, Yang Zeng, Qiang Luo, Wencong Zeng, Ruiming Tang, Kun Gai, et al. 2025. OneLoc: Geo-aware generative recommender systems for local life service. arXiv:2508.14646. Retrieved from https://arxiv.org/abs/2508.14646

[187] Mark Wilhelm, Ajith Ramanathan, Alexander Bonomo, Sagar Jain, Ed. H. Chi, and Jennifer Gillenwater. 2018. Practical diversified recommendations on youtube with determinantal point processes. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2165–2173.

[188] Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8 (1992), 229–256.

[189] Bin Wu, Feifan Yang, Zhangming Chan, Yu-Ran Gu, Jiawei Feng, Chao Yi, Xiang-Rong Sheng, Han Zhu, Jian Xu, Mang Ye, et al. 2025. MUSE: A simple yet effective multimodal search-based framework for lifelong user interest modeling. arXiv:2512.07216. Retrieved from https://arxiv.org/abs/2512.07216

[190] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2024. A survey on large language models for recommendation. *World Wide Web* 27, 5 (2024), 60.

[191] Xue Xia, Pong Eksombatchai, Nikil Pancha, Dhruvil Deven Badani, Po-Wei Wang, Neng Gu, Saurabh Vishwas Joshi, Nazanin Farahpour, Zhiyuan Zhang, and Andrew Zhai. 2023. Transact: Transformer-based realtime user action model for recommendation at pinterest. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 5249–5259.

[192] Xue Xia, Saurabh Vishwas Joshi, Kousik Rajesh, Kangnan Li, Yangyi Lu, Nikil Pancha, Dhruvil Deven Badani, Jiajing Xu, and Pong Eksombatchai. 2025. TransAct V2: Lifelong user action sequence modeling on pinterest recommendation. arXiv:2506.02267. Retrieved from https://arxiv.org/abs/2506.02267

[193] Bencheng Yan, Shilei Liu, Zhiyuan Zeng, Zihao Wang, Yizhen Zhang, Yujin Yuan, Langming Liu, Jiaqi Liu, Di Wang, Wenbo Su, et al. 2025. Unlocking scaling law in industrial recommendation systems with a three-step paradigm based large user model. arXiv:2502.08309. Retrieved from https://arxiv.org/abs/2502.08309

[194] Bencheng Yan, Pengjie Wang, Kai Zhang, Feng Li, Hongbo Deng, Jian Xu, and Bo Zheng. 2022. APG: Adaptive parameter generation network for click-through rate prediction. In *Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS '22)*, 24740–24752.

[195] Hao Yan, Shuai Ding, and Torsten Suel. 2009. Inverted index compression and query processing with optimized document ordering. In *Proceedings of the 18th International Conference on World Wide Web*, 401–410.

[196] Jing Yan, Liu Jiang, Jianfei Cui, Zhichen Zhao, Xingyan Bin, Feng Zhang, and Zuotao Liu. 2024. Trinity: Syncretizing multi-/long-tail/long-term interests all in one. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 6095–6104.

[197] Le Yan, Zhen Qin, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2022. Scale calibration of deep ranking models. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 4300–4309.

[198] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. arXiv:2505.09388. Retrieved from https://arxiv.org/abs/2505.09388

[199] Haoqiang Yang, Congde Yuan, Kun Bai, Mengzhuo Guo, Wei Yang, and Chao Zhou. 2025. HIT model: A hierarchical interaction-enhanced two-tower model for pre-ranking systems. arXiv:2505.19849. Retrieved from https://arxiv.org/abs/2505.19849

[200] Ji Yang, Xinyang Yi, Derek Zhiyuan Cheng, Lichan Hong, Yang Li, Simon Xiaoming Wang, Taibai Xu, and Ed. H. Chi. 2020. Mixed negative sampling for learning two-tower neural networks in recommendations. In *Companion Proceedings of the Web Conference 2020*, 441–447.

[201] Jia-Qi Yang, Xiang Li, Shuguang Han, Tao Zhuang, De-Chuan Zhan, Xiaoyi Zeng, and Bin Tong. 2021. Capturing delayed feedback in conversion rate prediction via elapsed-time sampling. *Proceedings of the AAAI Conference on Artificial Intelligence* 35 (2021), 4582–4589.

[202] Xiaoyong Yang, Yadong Zhu, Yi Zhang, Xiaobo Wang, and Quan Yuan. 2020. Large scale product graph construction for recommendation in e-commerce. arXiv:2010.05525. Retrieved from https://arxiv.org/abs/2010.05525

[203] Yuhao Yang, Chao Huang, Lianghao Xia, Yuxuan Liang, Yanwei Yu, and Chenliang Li. 2022. Multi-behavior hypergraph-enhanced transformer for sequential recommendation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2263–2274.

[204] Yuhao Yang, Zhi Ji, Zhaopeng Li, Yi Li, Zhonglin Mo, Yue Ding, Kai Chen, Zijian Zhang, Jie Li, Shuanglong Li, et al. 2025. Sparse meets dense: Unified generative recommendations with cascaded sparse-dense representations. arXiv:2503.02453. Retrieved from https://arxiv.org/abs/2503.02453

[205] Zhiming Yang, Haining Gao, Dehong Gao, Luwei Yang, Libin Yang, Xiaoyan Cai, Wei Ning, and Guannan Zhang. 2024. MLoRA: Multi-domain low-rank adaptive network for CTR prediction. In *Proceedings of the 18th ACM Conference on Recommender Systems*, 287–297.

[206] Shaowei Yao, Jiwei Tan, Xi Chen, Keping Yang, Rong Xiao, Hongbo Deng, and Xiaojun Wan. 2021. Learning a product relevance model from click-through data in e-commerce. In *Proceedings of the Web Conference 2021*, 2890–2899.

[207] Shaowei Yao, Jiwei Tan, Xi Chen, Juhao Zhang, Xiaoyi Zeng, and Keping Yang. 2022. ReprBERT: Distilling BERT to an efficient representation-based relevance model for e-commerce. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 4363–4371.

[208] Shota Yasui, Gota Morishita, Fujita Komei, and Masashi Shibata. 2020. A feedback shift correction in predicting conversion rates under delayed feedback. In *Proceedings of the Web Conference 2020*, 2740–2746.

[209] Dezhi Ye, Jie Liu, Junwei Hu, Jiabin Fan, Bowen Tian, Haijin Liang, and Jin Ma. 2025. Applying large language model for relevance search In tencent. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 5171–5181.

[210] Xinyang Yi, Ji Yang, Lichan Hong, Derek Zhiyuan Cheng, Lukasz Heldt, Aditee Kumthekar, Zhe Zhao, Li Wei, and Ed Chi. 2019. Sampling-bias-corrected neural modeling for large corpus item recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems*, 269–277.

[211] Dawei Yin, Yuening Hu, Jiliang Tang, Tim Daly, Mianwei Zhou, Hua Ouyang, Jianhui Chen, Changsung Kang, Hongbo Deng, Chikashi Nobata, et al. 2016. Ranking relevance in yahoo search. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 323–332.

[212] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 974–983.

[213] Yantao Yu, Weipeng Wang, Zhoutian Feng, and Daiyue Xue. 2021. A dual augmented two-tower model for online large-scale recommendation. In *Proceedings of the 3rd Workshop on Deep Learning Practice for High-Dimensional Sparse Data with KDD*.

[214] Wen Zan, Yaopeng Han, Xiaotian Jiang, Yao Xiao, Yang Yang, Dayao Chen, and Sheng Chen. 2023. SPM: Structured pretraining and matching architectures for relevance modeling in meituan search. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 4923–4929.

[215] Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Michael He, et al. 2024. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations. arXiv:2402.17152. Retrieved from https://arxiv.org/abs/2402.17152

[216] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1503–1512.

[217] Buyun Zhang, Liang Luo, Yuxin Chen, Jade Nie, Xi Liu, Shen Li, Yanli Zhao, Yuchen Hao, Yantao Yao, Ellie Dingqiao Wen, et al. 2024. Wukong: Towards a scaling law for large-scale recommendation. In *Proceedings of the 41st International Conference on Machine Learning*, 59421–59434.

[218] Buyun Zhang, Liang Luo, Xi Liu, Jay Li, Zeliang Chen, Weilin Zhang, Xiaohan Wei, Yuchen Hao, Michael Tsang, Wenjun Wang, et al. 2022. DHEN: A deep and hierarchical ensemble network for large-scale click-through rate prediction. arXiv:2203.11014. Retrieved from https://arxiv.org/abs/2203.11014

[219] Chao Zhang, Shiwei Wu, Haoxin Zhang, Tong Xu, Yan Gao, Yao Hu, and Enhong Chen. 2024. Notellm: A retrievable large language model for note recommendation. In *Companion Proceedings of the ACM Web Conference 2024*, 170–179.

[220] Chao Zhang, Haoxin Zhang, Shiwei Wu, Di Wu, Tong Xu, Xiangyu Zhao, Yan Gao, Yao Hu, and Enhong Chen. 2025. NoteLLM-2: Multimodal large representation models for recommendation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2815–2826.

[221] Daoze Zhang, Chenghan Fu, Zhanheng Nie, Jianyu Liu, Wanxian Guan, Yuan Gao, Jun Song, Pengjie Wang, Jian Xu, and Bo Zheng. 2025. MOON: Generative MLLM-based multimodal representation learning for e-commerce product understanding. arXiv:2508.11999. Retrieved from https://arxiv.org/abs/2508.11999

[222] Han Zhang, Songlin Wang, Kang Zhang, Zhiling Tang, Yunjiang Jiang, Yun Xiao, Weipeng Yan, and Wen-Yun Yang. 2020. Towards personalized and semantic retrieval: An end-to-end solution for e-commerce search via embedding learning. arXiv:2006.02282. Retrieved from https://arxiv.org/abs/2006.02282

[223] Jun Zhang, Yi Li, Yue Liu, Changping Wang, Yuan Wang, Yuling Xiong, Xun Liu, Haiyang Wu, Qian Li, Enming Zhang, et al. 2025. GPR: Towards a generative pre-trained one-model paradigm for large-scale advertising recommendation. arXiv:2511.10138. Retrieved from https://arxiv.org/abs/2511.10138

[224] Pengtao Zhang and Junlin Zhang. 2023. MemoNet: Memorizing all cross features' representations efficiently via multi-hash codebook network for CTR prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 3154–3163.

[225] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys* 52, 1 (2019), 1–38.

[226] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W. Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th Acm International Conference on Information and Knowledge Management*, 177–186.

[227] Zixing Zhang, Jürgen Geiger, Jouni Pohjalainen, Amr El-Desoky Mousa, Wenyu Jin, and Björn Schuller. 2018. Deep learning for environmentally robust speech recognition: An overview of recent developments. *ACM Transactions on Intelligent Systems and Technology* 9, 5 (2018), 1–28.

[228] Zhixuan Zhang, Yuheng Huang, Dan Ou, Sen Li, Longbin Li, Qingwen Liu, and Xiaoyi Zeng. 2023. Rethinking the role of pre-ranking in large-scale e-commerce searching system. arXiv:2305.13647. Retrieved from https://arxiv.org/abs/2305.13647

[229] Zhaoqi Zhang, Haolei Pei, Jun Guo, Tianyu Wang, Yufei Feng, Hui Sun, Shaowei Liu, and Aixin Sun. 2025. OneTrans: Unified feature interaction and sequence modeling with one transformer in industrial recommender. arXiv:2510.26104. Retrieved from https://arxiv.org/abs/2510.26104

[230] Cheng Zhao, Chenliang Li, Rong Xiao, Hongbo Deng, and Aixin Sun. 2020. CATN: Cross-domain recommendation for cold-start users via aspect transfer network. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 229–238.

[231] Gang Zhao, Ximing Zhang, Chenji Lu, Hui Zhao, Tianshu Wu, Pengjie Wang, Jian Xu, and Bo Zheng. 2024. Explainable LLM-driven multi-dimensional distillation for e-commerce relevance learning. arXiv:2411.13045. Retrieved from https://arxiv.org/abs/2411.13045

[232] Jun Zhao, Zhou Zhou, Ziyu Guan, Wei Zhao, Wei Ning, Guang Qiu, and Xiaofei He. 2019. IntentGC: A scalable graph convolution framework fusing heterogeneous information for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2347–2357.

[233] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. arXiv:2303.18223. Retrieved from https://arxiv.org/abs/2303.18223

[234] Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2018. Deep reinforcement learning for page-wise recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems*, 95–103.

[235] Zihuai Zhao, Wenqi Fan, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Zhen Wen, Fei Wang, Xiangyu Zhao, Jiliang Tang, et al. 2024. Recommender systems in the era of large language models (LLMs). *IEEE Transactions on Knowledge and Data Engineering* 36, 11 (2024), 6889–6907.

[236] Zhishan Zhao, Jingyue Gao, Yu Zhang, Shuguang Han, Siyuan Lou, Xiang-Rong Sheng, Zhe Wang, Han Zhu, Yuning Jiang, Jian Xu, et al. 2023. COPR: Consistency-oriented pre-ranking for online advertising. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 4974–4980.

[237] 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.

[238] Yukun Zheng, Jiang Bian, Guanghao Meng, Chao Zhang, Honggang Wang, Zhixuan Zhang, Sen Li, Tao Zhuang, Qingwen Liu, and Xiaoyi Zeng. 2022. Multi-objective personalized product retrieval in Taobao Search. arXiv:2210.04170. Retrieved from https://arxiv.org/abs/2210.04170

[239] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (2019), 5941–5948.

[240] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1059–1068.

[241] Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Filter-enhanced MLP is all you need for sequential recommendation. In *Proceedings of the ACM Web Conference*, 2388–2399.

[242] Han Zhu, Daqing Chang, Ziru Xu, Pengye Zhang, Xiang Li, Jie He, Han Li, Jian Xu, and Kun Gai. 2019. Joint optimization of tree-based index and deep model for recommender systems. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 3971–3980.

[243] Han Zhu, Junqi Jin, Chang Tan, Fei Pan, Yifan Zeng, Han Li, and Kun Gai. 2017. Optimized cost per click in taobao display advertising. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2191–2200.

[244] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1079–1088.

[245] Jie Zhu, Zhifang Fan, Xiaoxie Zhu, Yuchen Jiang, Hangyu Wang, Xintian Han, Haoran Ding, Xinmin Wang, Wenlin Zhao, Zhen Gong, et al. 2025. RankMixer: Scaling up ranking models in industrial recommenders. arXiv:2507.15551. Retrieved from https://arxiv.org/abs/2507.15551

[246] Tao Zhuang, Wenwu Ou, and Zhirong Wang. 2018. Globally optimized mutual influence aware ranking in e-commerce search. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 3725–3731.

[247] Jingwei Zhuo, Ziru Xu, Wei Dai, Han Zhu, Han Li, Jian Xu, and Kun Gai. 2020. Learning optimal tree models under beam search. In *Proceedings of the International Conference on Machine Learning*. PMLR, 11650–11659.

[248] Lixin Zou, Long Xia, Zhuoye Ding, Jiaxing Song, Weidong Liu, and Dawei Yin. 2019. Reinforcement learning to optimize long-term user engagement in recommender systems. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2810–2818.

[249] Lixin Zou, Long Xia, Pan Du, Zhuo Zhang, Ting Bai, Weidong Liu, Jian-Yun Nie, and Dawei Yin. 2020. Pseudo Dyna-Q: A reinforcement learning framework for interactive recommendation. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, 816–824.

[250] Lixin Zou, Long Xia, Yulong Gu, Xiangyu Zhao, Weidong Liu, Jimmy Xiangji Huang, and Dawei Yin. 2020. Neural interactive collaborative filtering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 749–758.

[251] 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 and Data Mining*, 4014–4022.

Received 7 August 2025; revised 27 December 2025; accepted 26 January 2026