Guaranteed Top-Adaptive-K in Recommendation

Nitin Bisht[1] , **Xiuwen Gong**[1*] , **Guandong Xu**[2*]
[1]University of Technology, Sydney
[2]The Education University of Hong Kong
Nitin.Bisht@student.uts.edu.au, Xiuwen.Gong@uts.edu.au, gdxu@eduhk.hk

Abstract

Recommender systems (RS) are crucial in offering personalized suggestions tailored to user preferences. While conventionally, Top-K recommendation approach is widely adopted, its reliance on fixed recommendation sizes overlooks diverse needs of users, leading to some relevant items not being recommended or vice versa. While recent work has made progress, they determine K by searching over all possible recommendation sizes for each user during inference. In real-world scenarios, with large datasets and numerous users with diverse and extensive preferences, this process becomes computationally impractical. Moreover, there is no theoretical guarantee of improved performance with the personalized K. In this paper, we propose a novel framework, Top-Adaptive-K, which determines dynamic K-prediction set size for each user efficiently and effectively. Generally, framework formulates recommendation problem within Conformal Risk Control paradigm and proposes the loss function based on user utility functions. A novel greedy optimization algorithm, K-Adapt, is designed to efficiently learn prediction sets. Theoretical analysis is provided to ensure recommendation performance by establishing upper bounds on the expected risk. Extensive experiments on multiple datasets demonstrate that Top-Adaptive-K framework outperforms baseline methods in both performance and time efficiency, offering guaranteed solution to fixed Top-K challenges.

1 Introduction

With the growing relevance of web as a medium for electronic and commercial transactions, Recommender Systems (RS) [Lu *et al.*, 2015; Aggarwal, 2016; Zhao *et al.*, 2023] have become ubiquitous for mitigating information overload, enabling platforms to deliver relevant suggestions across various domains such as e-commerce [Gulzar *et al.*, 2023], entertainment [Perano *et al.*, 2021], and job matching [Islam *et al.*, 2021]. They rank items based on users' preferences and their historical behaviors [Khatwani and Chandak, 2016; Cui *et al.*, 2020], thereby presenting the Top-K items [Cremonesi *et al.*, 2010; Li *et al.*, 2020], with the ranking scores, sorted in descending order. While this heuristic Top-K recommendation approach is widely adopted in the literature for its simplicity, a fundamental limitation is often overlooked: its reliance on a fixed K. This approach assumes that the same recommendation size will suffice for all users, ignoring their diverse needs and, leading to some relevant items not being recommended or vice versa. As a result, poor recommendation performance across users can lead to dissatisfaction and disengagement from the platform [Chen *et al.*, 2022].

For example, Figure 1 illustrates how NeuMF's oracle performance on Last.fm dataset, obtained by dynamically selecting each user's best per-metric set size (capped at 25) differs substantially from the performance under a single, fixed k (where fixed k is derived by averaging the user-specific (oracle) set sizes across the dataset for each metric) highlighting the inherent weakness of fixed-K recommendations.

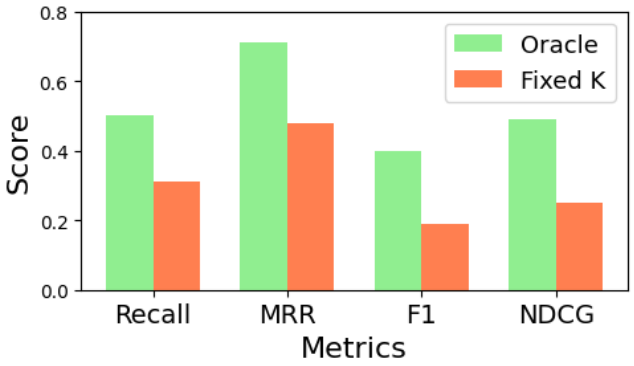

Figure 1: Performance Comparison of Oracle and Fixed K Across Metrics on Last.fM dataset using NeuMF.

While the concept of dynamically tailoring K to individual users is promising across multiple recommendation settings like optimizing screen space [Xi *et al.*, 2023], balancing user engagement and budget constraints [Chen *et al.*, 2022], or reducing user overload, existing research in this direction remains very limited. Recently, [KWEON *et al.*, 2024] modeled user-item interactions using Bernoulli distri-

*Corresponding author

butions during inference and approximated utility over ranked lists with Poisson-Binomial distribution to determine optimal K for each user. However, evaluating utility during inference is computationally impractical for real-world systems handling extra high dimension K, millions of users, and numerous preferences. This challenge also shares parallels with document list truncation methods [Wu *et al.*, 2021; Wang *et al.*, 2022]. However, these methods are prone to overfitting and poor generalization in the sparse and noisy contexts of recommendation datasets. Moreover, none of these methods provide statistical guarantees for model performance, which is essential for trustworthy recommendations.

Motivated by the above-mentioned challenges and taking inspiration from Conformal Prediction (CP) [Schafer *et al.*, 1999; Vovk *et al.*, 2005; Fontana *et al.*, 2023], we aim to propose a statistically sound user-tailored framework that incorporates uncertainty quantification into RS ecosystem. Specifically, our framework aims to achieve two key objectives: 1) dynamically and efficiently determining prediction set size for each user while 2) statistically ensuring that these dynamically generated prediction set sizes meet the desired performance guarantees across the dataset. CP, in its natural form, is designed to control and provide coverage guarantees. However, performance in RS is also evaluated through additional factors such as ranking quality and user satisfaction. This raises an important question: how can we adapt CP to conformalize different evaluation metrics in RS?

To address these challenges, we propose dataset and model-agnostic statistical framework- **Top-Adaptive-K** within the Conformal Risk Control paradigm [Bates *et al.*, 2021a; Angelopoulos *et al.*, 2024]. Specifically, the framework dynamically determines personalized K for each user by leveraging performance of the base recommendation model, ensuring statistically guaranteed performance across diverse recommender metrics. Our contributions are as follows: 1) We first propose a novel framework-Top-Adaptive-K which constructs loss functions based on utility functions tailored to key RS performance metrics and reformulates the recommendation problem within the conformal risk paradigm. 2) Secondly, we develop a novel and light-weight greedy-based optimization algorithm-**K-Adapt** to efficiently control the utility risk defined by Top-Adaptive-K below a user-defined threshold. 3) Next, we provide a rigorous theoretical analysis establishing the upper bound on expected risk. 4) Finally, we conduct extensive experiments across multiple datasets and metrics to demonstrate the effectiveness of Top-Adaptive-K in both performance as well as time efficiency.

2 Related Works

Personalized Recommendation Size. Traditional RS typically generate a fixed-size top-K list for each user [Yang *et al.*, 2012; Kang *et al.*, 2022; Li *et al.*, 2024]. A more recent direction involves dynamically adapting the recommendation size based on user preferences, analogous to document truncation in information retrieval [Arampatzis *et al.*, 2009; Wu *et al.*, 2021]. Methods like AttnCut and MtCut [Bahri *et al.*, 2020; Wang *et al.*, 2022] frame this as a classification task, predicting cutoff positions with a K-dimensional target. PerK [KWEON *et al.*, 2024] estimates expected user utility using calibrated interaction probabilities to personalize list sizes. However, unlike broader trends in machine learning (ML) [Gong *et al.*, 2021; Gong *et al.*, 2023; Gong *et al.*, 2024], these approaches lack probablistic modelling, leading to no guarantees in their performances.

Conformal Prediction (CP) CP [Papadopoulos *et al.*, 2002; Shafer and Vovk, 2008; Angelopoulos and Bates, 2022] provides distribution-free, finite-sample guarantees by constructing prediction sets satisfying $P(Y \notin C(X)) \leq \alpha$. They can help in uncertainty quantification by creating generalized bounds, as seen in other ML paradigms ([Liu and Tsang, 2017; Zou and Liu, 2023; Liu *et al.*, 2019]. Extensions address distribution shifts [Tibshirani *et al.*, 2019] and provide risk bounds beyond coverage [Bates *et al.*, 2021b]. Conformal risk control [Angelopoulos *et al.*, 2024] generalizes the approach to control monotone loss expectations supporting applications like false negative rate in multilabel tasks.

3 Preliminaries

We begin by introducing the notations used in this paper. Consider m items, denoted as $\boldsymbol{i} = [i]_{j=1}^{m}$, where each item i_j is an element of the item space $\mathcal{I}$. Similarly, we have n users, represented by $\boldsymbol{u} = [u]_{k=1}^{n}$, where each user u_k belongs to the user space $\mathcal{U}$. For brevity, we use u and i to denote a user and an item, respectively.

We focus on recommendations with implicit feedback [He *et al.*, 2016; Zhu *et al.*, 2024], a widely adopted scenario in RS. For a pair (u, i), an interaction label $Y_{u,i}$ is assigned a value of 1 if the interaction is observed, and 0 otherwise. Note that when $Y_{u,i} = 0$, it indicates that item i may either be irrelevant to user u or hidden-relevant item.

A dataset $\mathcal{D} = \{(u, i) \mid Y_{u,i} = 1\}$ consists of observed positive pairs and is partitioned into three mutually exclusive subsets: training ($\mathcal{D}_{\text{train}}$), calibration ($\mathcal{D}_{\text{calib}}$), and testing ($\mathcal{D}_{\text{test}}$). For user u, unobserved itemset $I_u^- = \{i \mid (u, i) \notin \mathcal{D}_{\text{train}}\}$ represents all items not observed in training set.

Prediction Sets After the recommender model $f_\theta : \mathcal{U} \times \mathcal{I} \to [0, 1]$ is trained on $\mathcal{D}_{\text{train}}$, we produce a ranked list $\pi(u)$ for the unobserved items in I_u^- by sorting their relevance scores $f_\theta(u, i)$ in descending order:

$$\pi(u) = \text{sort}_{i \in I_u^-} f_\theta(u, i), \tag{1}$$

where $\pi(u)$ represents the ranked order of unobserved items based on the estimated scores.

Top-K Predictions Traditionally, recommender systems generate *Top-K predictions* for a given user u by selecting the K-most relevant items from the ranked list $\pi(u)$:

$$\pi_K(u) = \pi(u)[: K], \tag{2}$$

where$[: K]$ denotes selecting the first K elements of the ranked list $\pi(u)$.

While this fixed-K approach is commonly adopted for simplicity, it fails to adapt to user-specific preferences and varying recommendation quality across users. This limitation motivates the exploration of dynamic prediction set sizes to better align recommendations with user needs.

4 The Proposed Framework

In this section, we develop our Top-Adaptive-K framework which creates personalized dynamic prediction set sizes for each user to ensure guaranteed performance across different recommendation metrics.

We begin by defining our set predictor $\phi_\lambda(u)$ dominated by the parameter λ to output calibrated prediction set $\pi_\lambda(u)$.

The calibrated prediction set $\pi_\lambda(u)$ is given by:

$$\pi_\lambda(u) = \Big\{ i \in \pi(u) \;\Big|\; f_\theta(u,i) \;\geq\; \lambda \Big\}. \tag{3}$$

The calibration is assumed to satisfy following property:

$$\lambda_1 < \lambda_2 \implies \pi_{\lambda_2}(u) \subseteq \pi_{\lambda_1}(u). \tag{4}$$

Next, to quantify alignment between prediction set $\pi_\lambda(u)$ and true relevant items $I_{\text{true}}(u)$, we introduce the concept of utility function, denoted as $U_{\text{M}}(I_{\text{true}}(u), \pi_\lambda(u))$. This function evaluates the quality of prediction set $\pi_\lambda(u)$ under some recommendation metric denoted by M, such as NDCG, thus measuring how well the set captures the user's true preferences.

Subsequently, we define the user-utility-based loss function for user u as follows:

$$L_u(\lambda) = 1 \;-\; U_M\big(I_{\text{true}}(u), \pi_\lambda(u)\big). \tag{5}$$

Now given Equation (5), we define the expected risk as:

$$R(\lambda) = \mathbb{E}_U[L_u(\lambda)]. \tag{6}$$

Following the framework of risk control [Angelopoulos *et al.*, 2024] and property in Equation (4), we attempt to find optimal λ^* to ensure probability of expected risk lower than α to be no less than confidence 1- η, which is formulated as:

$$\lambda^* = \sup\{\lambda \in \Lambda : \Pr(R(\lambda) \leq \alpha) \geq 1 - \eta\}. \tag{7}$$

Since, true data distribution is unknown, we use empirical risk $\hat{R}(\lambda)$ to approximate the expected risk $R(\lambda)$, given by:

$$\hat{R}(\lambda) = \frac{1}{n} \sum_{i=1}^{n} L_{u_i}(\lambda). \tag{8}$$

To this end, we complete the modeling of the proposed framework. To output the dynamic prediction sets for each user by Top-Adaptive-K such that the performance guarantee is met, we design a novel algorithm- K-Adapt based on a greedy strategy to obtain the parameter λ^*.

The K-Adapt framework is depicted in Figure 2 and the complete procedure of constructing dynamic prediction sets is summarized in Algorithm 1.

Prediction Set Construction: After obtaining λ^* from Algorithm 1, we construct prediction sets for $\mathcal{D}_{\text{test}}$. For each user u in test dataset, we create the prediction set by selecting items with relevance scores greater than λ^*. The prediction sets, tailored to individual user preferences, are ensured to control the risk below the user-defined risk threshold α.

† $Here, \Delta(\lambda)$ is equivalent to $\frac{\lambda}{|\Lambda|}$, where Λ is set of λ.

Algorithm 1 K-Adapt Algorithm

Input: Recommendation model $f_\theta(u,i)$, calibration dataset $\mathcal{D}_{calib}$, initial control parameter λ_{init}, user-defined risk threshold α, error tolerance ϵ.
Output: Optimal λ^* .

```
Define utility function U_M(·) based on the recommendation metric.
Goal: Find the λ* that meets the risk guarantee as in Equation (7).
Initialize λ ← λ_init.
while λ < 1 do
    Generate π_λ(u) using f_θ(u,i) > λ based on Equation (3) .
    Calculate L_u(λ) using Equation (5).
    Calculate R̂(λ) based on Equation (8).
    if R̂(λ) ≤ α − ε then
        return λ
    else
        Update λ ← λ − Δλ†
    end if
end while
```

4.1 Utility Functions

The design of our loss function in Equation (5) makes the proposed framework more flexible by accommodating different types of recommender utility functions. Below, we describe the utility functions for commonly used metrics:

1. Utility for Recall:

$$U_{\text{recall}}(I_{\text{true}}(u), \pi_\lambda(u)) = \frac{|I_{\text{true}}(u) \cap \pi_\lambda(u)|}{|I_{\text{true}}(u)|}. \tag{9}$$

This utility measures proportion of relevant items included in prediction set. Here, $|\cdot|$ denotes the cardinality of a set.

2. Utility for Mean Reciprocal Rank:

$$U_{\text{mrr}}(I_{\text{true}}(u), \pi_\lambda(u)) = \begin{cases} \frac{1}{\min\left\{ r(i) \;\middle|\; i \in I_{\text{true}}(u) \cap \pi_\lambda(u) \right\}}, \\ \quad \text{if } I_{\text{true}}(u) \cap \pi_\lambda(u) \neq \emptyset, \\ 0, \quad \text{otherwise.} \end{cases} \tag{10}$$

This utility measures how early the first relevant item appears in the ranked list. The term $r(i)$ represents the rank of item i in the prediction set $\pi_\lambda(u)$.

3. Utility for F1-Score :

$$U_{\text{F1}}(I_{\text{true}}(u), \pi_\lambda(u)) = \frac{2|I_{\text{true}}(u) \cap \pi_\lambda(u)|}{|I_{\text{true}}(u)| + |\pi_\lambda^{\max}(u)|}. \tag{11}$$

This utility balances how many of the relevant items are actually predicted with how many of the predicted items are truly relevant. Here, $|\pi_\lambda^{\max}(u)|$ is the maximum possible size of the prediction set for user u at the given threshold λ.

4. Utility for NDCG :

$$U_{\text{ndcg}}(I_{\text{true}}(u), \pi_\lambda(u)) = \frac{\sum_{i=1}^{|\pi_\lambda(u)|} \frac{\mathbb{I}[i \in I_{\text{true}}(u)]}{\log_2(i+1)}}{\sum_{i=1}^{|I_{\text{true}}(u)|} \frac{1}{\log_2(i+1)}}. \tag{12}$$

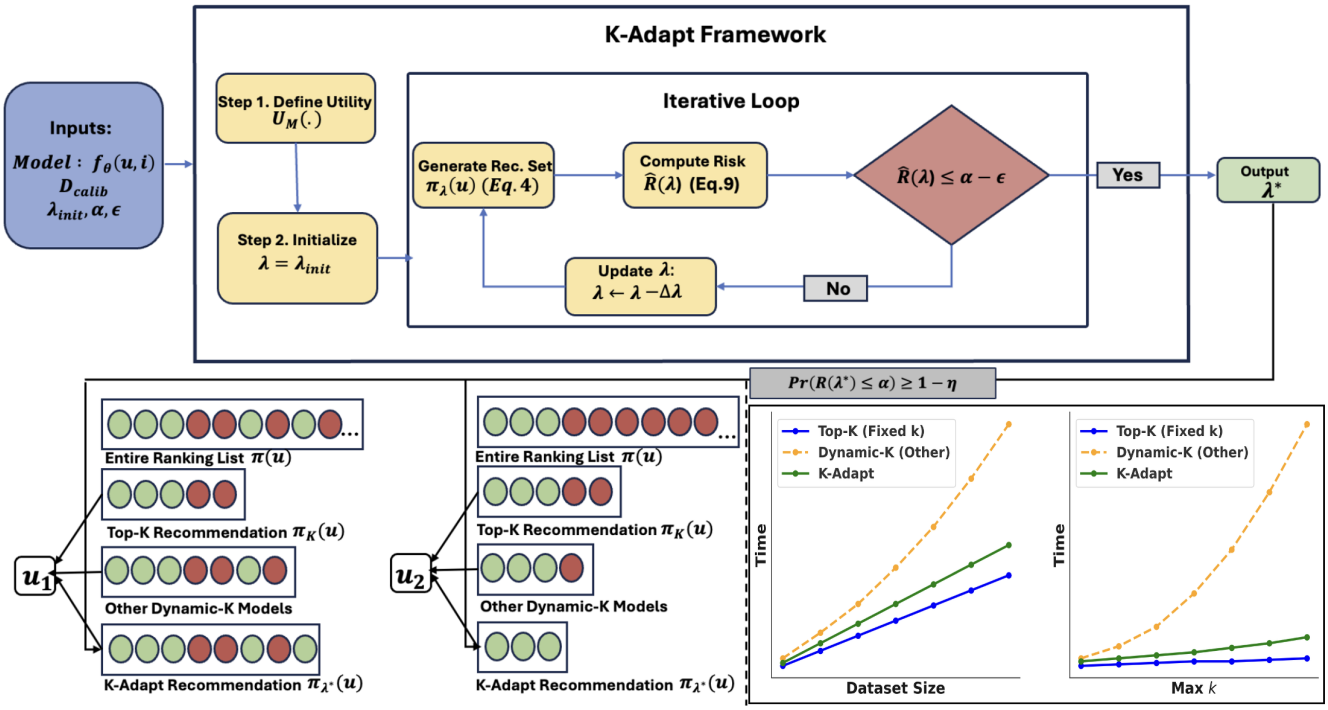

Figure 2: **The K-Adapt Framework.** The *top* portion outlines how λ is calibrated to choose recommendation list lengths under a specified risk constraint. In the **lower left**, we compare final recommendations from top-K, other dynamic-K, and K-Adapt (green = relevant, red = irrelevant), showing how K-Adapt flexibly selects up to a maximum allowable length (max k) while maintaining high relevance. The **lower right** plots runtime versus dataset size and max k, illustrating that K-Adapt's computational overhead remains manageable even as the dataset size and max k grows, providing superior performance with time efficiency.

This utility measures ranking quality by assigning higher importance to relevant items appearing earlier in ranked list. Here, $\mathbb{I}[i \in I_{\text{true}}(u)]$ is an indicator function that returns 1 if item i is relevant, and 0 otherwise. The term $\log_2(i+1)$ is a position-based discount factor to penalize items ranked lower.

By leveraging utility functions tailored to specific metrics, Top-Adaptive-K accommodates a wide range of evaluation criteria such as Recall, MRR, F1, and NDCG.

5 Theoretical Analysis

In the previous sections, we demonstrate how our framework utilizes trained model $f_\theta(u,i)$ and a calibration dataset $\mathcal{D}_{\text{calib}}$ to compute λ^*, which generates a dynamic recommendation set for each user during testing. However, it remains to be established whether λ^* learned from the empirical risk can statistically guarantee the expected risk control below the user-defined threshold. In this section, we theoretically derive the upper bounds of expected risk to show how it can be approximately controlled by the threshold α with the learned λ^*.

To formalize theoretical guarantee, we use Hoeffding's inequality, which enables uniform probabilistic control over the deviations between empirical and expected risks [Bentkus, 2004]. By leveraging the finite nature of Λ, which is the set of λ values, we apply a union bound to establish that the risk deviation remains bounded with high probability. This result has been depicted in the following theorem:

Theorem 1 (Expected Risk Control)**.** *Let Λ be a finite set of λ. The expected risk $R(\lambda)$ is right-continuous in λ, and is bounded within $[0, B]$ for some $B > 0$ and all u and λ.*

For any $\eta > 0$, let $\delta(\epsilon)$ be the distribution deviation between the expected and empirical risk such that it satisfies: $2|\Lambda| \exp\left(-\frac{2n\,\delta(\epsilon)^2}{B^2}\right) \le \eta$. Then, for $\lambda^* \in \Lambda$, with probability at least $1-\eta$ we have,

$$R(\lambda^*) \le \alpha + \frac{B}{\sqrt{2n}} \sqrt{\ln\left(\frac{2|\Lambda|}{\eta}\right)}.$$

Proof. We first consider the empirical risk on $n+1$ samples and using eq. (8):

$$\hat{R}_{n+1}(\lambda) = \frac{n}{n+1}\hat{R}(\lambda) + \frac{1}{n+1} L_{u_{n+1}}(\lambda). \tag{13}$$

where $\hat{R}(\lambda)$ is empirical risk based on the first n samples, and $L_{u_{n+1}}(\lambda)$ is loss on the $(n+1)$th sample.

For $\epsilon > 0$, suppose $\hat{R}(\lambda^*) \le \alpha - \epsilon$., then from (13):

$$\hat{R}_{n+1}(\lambda^*) \le \frac{n}{n+1}(\alpha - \epsilon) + \frac{1}{n+1} L_{u_{n+1}}(\lambda^*).$$

Multiplying both sides by $(n+1)$ and rearranging:

$$(n+1)\,\hat{R}_{n+1}(\lambda^*) \le n(\alpha - \epsilon) + L_{u_{n+1}}(\lambda^*),$$

$$L_{u_{n+1}}(\lambda^*) \ge (n+1)\,\hat{R}_{n+1}(\lambda^*) - n(\alpha - \epsilon).$$

Taking expectations on both sides, we get:

$$\mathbb{E}\big[L_{u_{n+1}}(\lambda^*)\big] \;\geq\; (n+1)\,\mathbb{E}\big[\hat{R}_{n+1}(\lambda^*)\big] \;-\; n\,(\alpha-\epsilon).$$

Since $\mathbb{E}[L_{u_{n+1}}(\lambda)] = R(\lambda)$, we have

$$R(\lambda^*) \;\geq\; (n+1)\,\mathbb{E}\big[\hat{R}_{n+1}(\lambda^*)\big] \;-\; n\,(\alpha-\epsilon).$$

Next, we relate $\hat{R}(\lambda)$ to $R(\lambda)$ through Hoeffding's inequality. For any fixed λ, Hoeffding's inequality states:

$$P\Big(\big|R(\lambda)-\hat{R}(\lambda)\big| > \delta\Big) \leq 2\exp\Big(-\tfrac{2n\,\delta^2}{B^2}\Big), \qquad (14)$$

where n is the number of samples, δ is deviation between expected risk and empirical risk and B is the bound of risk. Since Λ is a finite set of thresholds, we apply union bound over all $\lambda \in \Lambda$. This results in:

$$P\Big(\exists\,\lambda\in\Lambda : \big|R(\lambda)-\hat{R}(\lambda)\big| > \delta\Big) \leq \sum_{\lambda\in\Lambda} P\Big(\big|R(\lambda)-\hat{R}(\lambda)\big| > \delta\Big) \leq 2|\Lambda|\exp\Big(-\frac{2n\,\delta^2}{B^2}\Big). \qquad (15)$$

Hence, equivalently we can say that:

$$P\Big(\sup_{\lambda\in\Lambda}\big|R(\lambda)-\hat{R}(\lambda)\big| \;\leq\; \delta\Big) \geq 1-2|\Lambda|\exp\Big(-\tfrac{2n\,\delta^2}{B^2}\Big).$$

We now choose $\delta = \delta(\epsilon)$ to ensure this event has probability at least $1-\eta$. Concretely, we set

$$2|\Lambda|\;\exp\Big(-\tfrac{2n\,\delta(\epsilon)^2}{B^2}\Big) \;=\; \eta,$$

implying that with probability at least $1-\eta$,

$$\sup_{\lambda\in\Lambda}\big|R(\lambda)-\hat{R}(\lambda)\big| \;\leq\; \delta(\epsilon).$$

In particular, for any specific $\lambda \in \Lambda$, $|R(\lambda)-\hat{R}(\lambda)| \;\leq\; \delta(\epsilon)$.

Let us return to λ^*. From $\hat{R}(\lambda^*) \leq \alpha-\epsilon$ and the bound $|R(\lambda^*)-\hat{R}(\lambda^*)| \leq \delta(\epsilon)$, we obtain:

$$R(\lambda^*) \;\leq\; \hat{R}(\lambda^*)+\delta(\epsilon) \;\leq\; (\alpha-\epsilon)+\delta(\epsilon).$$

Since ϵ can be arbitrarily small, we typically write

$$R(\lambda^*) \;\leq\; \alpha+\delta(\epsilon).$$

Finally, plugging $\delta(\epsilon) \;=\; \frac{B}{\sqrt{2n}}\sqrt{\ln\Big(\frac{2|\Lambda|}{\eta}\Big)}$ into the above completes the proof that with probability at least $1-\eta$:

$$R(\lambda^*) \;\leq\; \alpha \;+\; \frac{B}{\sqrt{2n}}\sqrt{\ln\Big(\tfrac{2|\Lambda|}{\eta}\Big)}$$

Hence Proved. □

Remark. *From Theorem 1, we can see the expected risk can be upper bounded by α and a constant term. When the sample size $n \to \infty$, $\frac{B}{\sqrt{2n}}\sqrt{\ln\left(\frac{2|\Lambda|}{\eta}\right)} \to 0$, the upper bound of the expected risk approaches to α.*

6 Experiments

In this section, we conduct experiments to validate the effectiveness of the proposed framework. We design experiments to 1) validate whether the framework can achieve superior performance by comparing it with the state-of-the-art baselines. 2) whether our method can achieve higher time efficiency compared to other baselines, and 3) analyze how the risk threshold α and the confidence parameter η influence the performance and the average optimal prediction set sizes.

6.1 Datasets and Baseline Methods

We experiment on three real-world datasets- MovieLens 100k (Movies) [McAuley *et al.*, 2015], Last.fM (Music) [Cantador *et al.*, 2011] and AmazonOffice (eCommerce) [Harper and Konstan, 2015].

To obtain relevance scores, we use five widely recognized recommender models representing diverse architectures: a) DeepFM [Guo *et al.*, 2017]; b) LightGCN [He *et al.*, 2020]; c) GMF [Koren *et al.*, 2009]; d) MLP [Zhang *et al.*, 2019] and e) NeuMF [He *et al.*, 2017]. To evaluate the effectiveness of K-Adapt, we compare it against the following baseline models:

- **Avg-K method:** Utilizes the average of the prediction set sizes returned by K-Adapt during calibration and uses it as a fixed k value for all users.
- **AttnCut[Wu *et al.*, 2021]:** Employs a Bi-LSTM and Transformer encoder in a classification framework to predict the optimal cutoff position in ranked lists.
- **MtCut [Wang *et al.*, 2022]:** Enhances AttnCut using Multi-gate Mixture-of-Experts (MMoE) model, leveraging multi-task learning for improved cutoff prediction.
- **PerK [KWEON *et al.*, 2024]:** Utilizes Poisson-Binomial approximation to compute the expected utility at each cutoff position in ranked lists.

6.2 Implementation Details

All base recommender models are trained using the Adam optimizer for 20 epochs with a learning rate of 0.001 and batch size of 256. The scores $f_\theta(u,i)$ generated by these models serve as inputs for the dynamic k methods. Baseline configurations are as follows: MtCut uses 3 experts with Transformer layers of size 128, 2 attention heads, a gating mechanism, and a bi-directional LSTM of size 64; AttnCut uses a Transformer layer of size 64 with 2 attention heads and a bi-directional LSTM of size 32; PerK employs a Poisson-Binomial approximation to compute expected utility values. Our framework, K-Adapt, uses a risk threshold $\alpha = 0.05$ and confidence parameter $\eta = 0.05$. Held-out training data is split into 60% calibration and 40% testing. We set a negative sampling rate of 50 per true interaction and cap recommendation size at 25 items per user. Evaluation is done against Oracle values by computing the optimal prediction set (the selection of items that maximizes the metric for each user). Code is publicly available at https://github.com/kalpiree/Top-Adaptive-K.

6.3 Experimental Results

We evaluate the performance of all methods, i.e. Avg-K, AttnCut, MtCut, PerK, and K-Adapt using Recall, MRR, F1,

BaseModel	Method	MovieLens				Last.fM				AmazonOffice			
		Recall	MRR	F1	NDCG	Recall	MRR	F1	NDCG	Recall	MRR	F1	NDCG
DeepFM	Oracle	0.47	0.72	0.37	0.47	0.47	0.70	0.38	0.46	0.52	0.32	0.23	0.28
	Avg-K	0.36	0.59	0.23	0.29	0.33	0.50	0.25	0.31	0.32	0.19	0.11	0.20
	AttnCut	0.38	0.62	0.29	0.32	0.32	0.58	0.31	0.35	0.33	0.14	0.15	0.16
	MtCut	0.39	0.61	0.29	0.32	0.36	0.58	0.31	0.38	0.37	0.15	0.15	0.17
	PerK	0.41	0.62	0.30	0.37	0.40	0.61	0.32	0.40	0.44	0.23	0.17	0.21
	K-Adapt (Ours)	**0.43**	**0.67**	**0.33**	**0.42**	**0.43**	**0.65**	**0.34**	**0.43**	**0.48**	**0.28**	**0.18**	**0.23**
LightGCN	Oracle	0.50	0.72	0.39	0.45	0.51	0.67	0.39	0.47	0.51	0.34	0.23	0.30
	Avg-K	0.35	0.59	0.25	0.33	0.41	0.52	0.30	0.37	0.42	0.23	0.13	0.20
	AttnCut	0.37	0.61	0.28	0.36	0.37	0.54	0.31	0.36	0.35	0.23	0.15	0.19
	MtCut	0.41	0.64	0.29	0.36	0.39	0.56	0.32	0.38	0.37	0.25	0.15	0.21
	PerK	0.43	0.62	0.30	0.38	0.43	0.50	0.32	0.36	0.47	0.27	0.16	0.23
	K-Adapt (Ours)	**0.45**	**0.68**	**0.34**	**0.40**	**0.47**	**0.64**	**0.34**	**0.42**	**0.48**	**0.29**	**0.18**	**0.25**
GMF	Oracle	0.41	0.67	0.32	0.38	0.46	0.61	0.37	0.45	0.47	0.28	0.21	0.28
	Avg-K	0.17	0.53	0.21	0.20	0.41	0.55	0.31	0.38	0.41	0.21	0.13	0.22
	AttnCut	0.29	0.58	0.25	0.31	0.26	0.52	0.32	0.35	0.31	0.20	0.12	0.19
	MtCut	0.31	0.60	0.24	0.33	0.27	0.54	0.32	0.37	0.35	0.20	0.12	0.21
	PerK	0.35	0.57	0.25	0.32	0.40	0.55	0.32	0.38	0.44	0.18	0.13	0.21
	K-Adapt (Ours)	**0.38**	**0.62**	**0.27**	**0.34**	**0.42**	**0.57**	**0.34**	**0.41**	**0.46**	**0.24**	**0.16**	**0.24**
MLP	Oracle	0.48	0.70	0.37	0.43	0.47	0.67	0.40	0.45	0.46	0.30	0.22	0.27
	Avg-K	0.25	0.61	0.23	0.30	0.24	0.44	0.19	0.23	0.37	0.16	0.11	0.14
	AttnCut	0.39	0.60	0.26	0.31	0.21	0.54	0.30	0.38	0.29	0.18	0.13	0.20
	MtCut	0.41	0.60	0.27	0.36	0.23	0.56	0.30	0.40	0.33	0.19	0.13	0.20
	PerK	0.41	0.61	0.29	0.38	0.33	0.57	0.34	0.39	0.41	0.18	0.15	0.21
	K-Adapt (Ours)	**0.44**	**0.66**	**0.33**	**0.40**	**0.43**	**0.63**	**0.37**	**0.42**	**0.43**	**0.26**	**0.16**	**0.24**
NeuMF	Oracle	0.51	0.74	0.39	0.47	0.50	0.71	0.40	0.49	0.50	0.31	0.23	0.30
	Avg-K	0.38	0.61	0.24	0.33	0.31	0.48	0.19	0.25	0.32	0.22	0.12	0.22
	AttnCut	0.40	0.63	0.25	0.34	0.35	0.55	0.32	0.39	0.34	0.22	0.15	0.20
	MtCut	0.40	0.64	0.26	0.38	0.38	0.57	0.34	0.40	0.37	0.24	0.14	0.19
	PerK	0.43	0.62	0.32	0.39	0.42	0.58	**0.36**	0.42	0.44	0.23	0.16	**0.25**
	K-Adapt (Ours)	**0.46**	**0.69**	**0.34**	**0.42**	**0.45**	**0.67**	**0.36**	**0.44**	**0.46**	**0.25**	**0.18**	**0.25**

Table 1: Performance comparison between K-Adapt (Ours) and baseline methods under various BaseModels (DeepFM, LighGCN, GMF, MLP, NeuMF), metrics (Recall, MRR, F1, NDCG) across different Datasets (MovieLens, Last.fM, AmazonOffice). For K-Adapt, α and η are set empirically as 0.05 respectively. Bold indicates best result and underline marks second best.

and NDCG across three datasets: MovieLens, LastFM, and AmazonOffice, implemented on five base recommendation models: DeepFM, LightGCN, GMF, MLP, and NeuMF. Detailed results are shown in Table 1, from which we make the following observations:

- The proposed K-Adapt framework consistently outperforms all baseline methods and cwith comparable results to Oracle performance across datasets and metrics, when controlling risk within $\alpha = 0.05$ and confidence $1 - \eta = 95\%$, aligning with theoretical expectations.

- Avg-K serves as a strong baseline but lags behind dynamic-k methods, especially on dense datasets like

Method	Average Time (in sec)		
	Movielens	LastFM	Amazon Office
AttnCut	125.67	601.67	905.88
MtCut	425.13	724.08	1017.15
PerK	1205.78	3905.78	7560.67
K-Adapt (Ours)	**24.08**	**55.27**	**94.28**

Table 2: Average Time (in sec) on Various Datasets

MovieLens, highlighting the need for personalized prediction sizes to optimize user satisfaction.

- Dynamic-k methods such as AttnCut and MtCut surpass Avg-K on dense datasets but degrade on sparser ones like LastFM and AmazonOffice due to overfitting because of their reliance on high-dimensional features (e.g., embeddings).
- PerK outperforms AttnCut and MtCut by modeling user-specific interaction likelihoods via the Bernoulli-Poisson framework, enabling better generalization across data densities. However, it is still inferior to K-Adapt due to its dependence on determining accurately calibrated interaction probabilities, which, despite user-wise calibration, may fail to fully adapt to the variability of user preferences in highly dynamic environments.
- Overall, results highlight K-Adapt's data- and model-agnostic design, achieving robust and superior performance across all metrics, base models, and datasets.

6.4 Time Efficiency Analysis

We analyze the computational cost (training time) of the Top-Adaptive-K framework compared to other dynamic-k methods. Results averaged across all BaseModels are shown in Table 2. depict that our framework is significantly more time-efficient, highlighting its scalability. This is because methods like AttnCut depend on neural architectures such as Bi-LSTM (h) and Transformer Encoder (d), leading to time complexity $\mathcal{O}(u \cdot n \cdot (h^2 + d^2))$, where u is the number of users and n the prediction set size. MtCut further increases complexity by incorporating e experts, leading to $\mathcal{O}(u \cdot n \cdot (h^2 + e \cdot d^2))$ due to added model parameters. PerK avoids neural networks and estimates utilities during calibration, making it lighter during calibration but computationally expensive at inference, especially when n or the candidate range of k (m) is large.

In contrast, our framework eliminates neural components and learns λ during calibration, avoiding runtime optimization over k. TAs a result, it has a time complexity of $\mathcal{O}(u \cdot n \log n)$, independent of m, making K-Adapt highly efficient for practical applications.

6.5 Parameter Analysis

We analyze the influence of parameters α (risk threshold) and η (confidence threshold) on the performance of the K-Adapt framework, focusing on metrics such as NDCG, F1 score, and average prediction set size.

Figure 3 reports the impact of risk threshold α varying from 0.10 to 0.50 (in increments of 0.05) on average prediction set sizes under fixed confidence threshold $\eta = 0.10$ using Last.fM dataset on Recall and MRR metrics respectively. We observe that as α increases, both average prediction set size and performance metrics exhibit a decreasing trend. This behavior aligns with theoretical expectation, as increasing α relaxes the risk threshold, allowing model to generate smaller prediction sets but at lower performance.

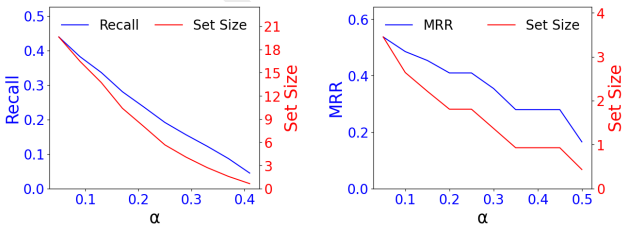

Figure 3: Performance trends on the **Last.fm** dataset with varying α and fixed $\eta = 0.1$.

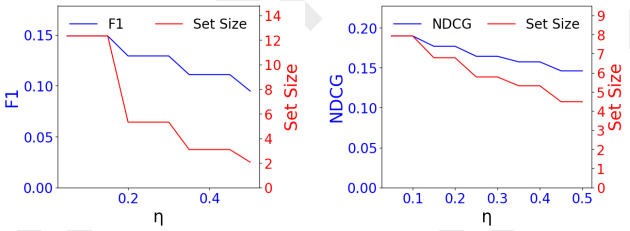

Figure 4: Performance trends on the **AmazonOffice** dataset with varying η and fixed $\alpha = 0.1$.

Figure 4 evaluates impact of confidence level η varying from 0.10 to 0.50 (in increments of 0.05) on average prediction set sizes under fixed risk threshold $\alpha = 0.10$ using AmazonOffice dataset on F1 and NDCG metrics respectively. We observe that when η increases, model becomes less conservative, leading to reduction in both prediction set size and performance metrics. This phenomenon demonstrates the framework's ability to balance between prediction set tightness and performance guarantees based on confidence threshold.

This analysis provides insights into prediction control, enabling practitioners to adjust prediction set size and performance according to desired risk and confidence levels. Additional plots showing similar trends are available in the code repository.

7 Conclusion

In this paper, we address limitations of fixed prediction set sizes in recommender systems, which can lead to user dissatisfaction. We propose Top-Adaptive-K, a dynamic framework that determines personalized set sizes via Conformal Risk Control with theoretical guarantees. Experiments show that Top-Adaptive-K outperforms heuristic dynamic-k methods, achieving strong performance with well-tuned risk (α) and confidence (η) levels. This work lays the groundwork for reliable, adaptive recommendation sets in diverse RS settings.

Acknowledgments

This work is partially supported by the Australian Research Council (ARC) Under Grants DP220103717 and LE220100078, and the National Natural Science Foundation of China under Grants No.62072257.

References

[Aggarwal, 2016] Charu C. Aggarwal. *Recommender Systems: The Textbook*. Springer Publishing Company, Incorporated, 1st edition, 2016.

[Angelopoulos and Bates, 2022] Anastasios N. Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification, 2022.

[Angelopoulos *et al.*, 2024] Anastasios Nikolas Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. Conformal risk control. In *The Twelfth International Conference on Learning Representations*, 2024.

[Arampatzis *et al.*, 2009] Avi Arampatzis, Jaap Kamps, and Stephen Robertson. Where to stop reading a ranked list? threshold optimization using truncated score distributions. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09. Association for Computing Machinery, 2009.

[Bahri *et al.*, 2020] Dara Bahri, Yi Tay, Che Zheng, Donald Metzler, and Andrew Tomkins. Choppy: Cut transformer for ranked list truncation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, New York, NY, USA, 2020. Association for Computing Machinery.

[Bates *et al.*, 2021a] Stephen Bates, Anastasios Angelopoulos, Lihua Lei, Jitendra Malik, and Michael Jordan. Distribution-free, risk-controlling prediction sets. *Journal of the ACM (JACM)*, 68(6):1–34, 2021.

[Bates *et al.*, 2021b] Stephen Bates, Anastasios Angelopoulos, Lihua Lei, Jitendra Malik, and Michael I. Jordan. Distribution-free, risk-controlling prediction sets, 2021.

[Bentkus, 2004] Vidmantas Bentkus. On hoeffding's inequalities. *The Annals of Probability*, 32(2), April 2004.

[Cantador *et al.*, 2011] Ivan Cantador, Peter Brusilovsky, and Tsvi Kuflik. Second workshop on information heterogeneity and fusion in recommender systems (hetrec2011). In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, page 387–388. Association for Computing Machinery, 2011.

[Chen *et al.*, 2022] Dagui Chen, Qi Yan, Chunjie Chen, Zhenzhe Zheng, Yangsu Liu, Zhenjia Ma, Chuan Yu, Jian Xu, and Bo Zheng. Hierarchically constrained adaptive ad exposure in feeds. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, page 3003–3012. Association for Computing Machinery, 2022.

[Cremonesi *et al.*, 2010] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10. Association for Computing Machinery, 2010.

[Cui *et al.*, 2020] Zhihua Cui, Xianghua Xu, XUE Fei, Xingjuan Cai, Yang Cao, Wensheng Zhang, and Jinjun Chen. Personalized recommendation system based on collaborative filtering for iot scenarios. *IEEE Transactions on Services Computing*, 2020.

[Fontana *et al.*, 2023] Matteo Fontana, Gianluca Zeni, and Simone Vantini. Conformal prediction: a unified review of theory and new challenges. *Bernoulli*, 2023.

[Gong *et al.*, 2021] Xiuwen Gong, Dong Yuan, and Wei Bao. Understanding partial multi-label learning via mutual information. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[Gong *et al.*, 2023] Xiuwen Gong, Dong Yuan, Wei Bao, and Fulin Luo. A unifying probabilistic framework for partially labeled data learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(7):8036–8048, 2023.

[Gong *et al.*, 2024] Xiuwen Gong, Nitin Bisht, and Guandong Xu. Does label smoothing help deep partial label learning? In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[Gulzar *et al.*, 2023] Yonis Gulzar, Ali A Alwan, Radhwan M Abdullah, Abedallah Zaid Abualkishik, and Mohamed Oumrani. Oca: ordered clustering-based algorithm for e-commerce recommendation system. *Sustainability*, 2023.

[Guo *et al.*, 2017] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: A factorization-machine based neural network for ctr prediction, 2017.

[Harper and Konstan, 2015] F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 2015.

[He *et al.*, 2016] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 549–558, 2016.

[He *et al.*, 2017] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering, 2017.

[He *et al.*, 2020] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. 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*, SIGIR '20, page 639–648. Association for Computing Machinery, 2020.

[Islam *et al.*, 2021] Rashidul Islam, Kamrun Naher Keya, Ziqian Zeng, Shimei Pan, and James Foulds. Debiasing career recommendations with neural fair collaborative filtering. In *Proceedings of the 2009 SIAM International Conference on Data Mining*, WWW '21, New York, NY, USA, 2021. Association for Computing Machinery.

[Kang *et al.*, 2022] SeongKu Kang, Dongha Lee, Wonbin Kweon, Junyoung Hwang, and Hwanjo Yu. Consensus learning from heterogeneous objectives for one-class collaborative filtering. In *Proceedings of the ACM Web Conference 2022*, pages 1965–1976, 2022.

[Khatwani and Chandak, 2016] Sneha Khatwani and MB Chandak. Building personalized and non personalized recommendation systems. In *2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*. IEEE, 2016.

[Koren *et al.*, 2009] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[KWEON *et al.*, 2024] WONBIN KWEON, SeongKu Kang, Sanghwan Jang, and Hwanjo Yu. Top-personalized-k recommendation. In *The Web Conference 2024*, 2024.

[Li *et al.*, 2020] Dong Li, Ruoming Jin, Jing Gao, and Zhi Liu. On sampling top-k recommendation evaluation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2114–2124, 2020.

[Li *et al.*, 2024] Jianing Li, Chaoqun Yang, Guanhua Ye, and Quoc Viet Hung Nguyen. Graph neural networks with deep mutual learning for designing multi-modal recommendation systems. *Information Sciences*, 2024.

[Liu and Tsang, 2017] Weiwei Liu and Ivor W. Tsang. Making decision trees feasible in ultrahigh feature and label dimensions. *J. Mach. Learn. Res.*, 18(1):2814–2849, January 2017.

[Liu *et al.*, 2019] Weiwei Liu, Xiaobo Shen, Bo Du, Ivor W. Tsang, Wenjie Zhang, and Xuemin Lin. Hyperspectral imagery classification via stochastic hhsvms. *IEEE Transactions on Image Processing*, 28(2):577–588, 2019.

[Lu *et al.*, 2015] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. Recommender system application developments: a survey. *Decision support systems*, 74:12–32, 2015.

[McAuley *et al.*, 2015] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes, 2015.

[Papadopoulos *et al.*, 2002] Harris Papadopoulos, Kostas Proedrou, Volodya Vovk, and Alexander Gammerman. Inductive confidence machines for regression. In *Proceedings of the 13th European Conference on Machine Learning*, ECML '02, page 345–356, Berlin, Heidelberg, 2002. Springer-Verlag.

[Perano *et al.*, 2021] Mirko Perano, Gian Luca Casali, Yulin Liu, and Tindara Abbate. Professional reviews as service: A mix method approach to assess the value of recommender systems in the entertainment industry. *Technological Forecasting and Social Change*, 2021.

[Schafer *et al.*, 1999] Ben Schafer, Joseph Konstan, and John Riedl. Recommender systems in e-commerce. *1st ACM Conference on Electronic Commerce, Denver, Colorado, United States*, 10 1999.

[Shafer and Vovk, 2008] Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9(3), 2008.

[Tibshirani *et al.*, 2019] Ryan J Tibshirani, Rina Foygel Barber, Emmanuel Candes, and Aaditya Ramdas. Conformal prediction under covariate shift. *Advances in neural information processing systems*, 32, 2019.

[Vovk *et al.*, 2005] Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer-Verlag, Berlin, Heidelberg, 2005.

[Wang *et al.*, 2022] Dong Wang, Jianxin Li, Tianchen Zhu, Haoyi Zhou, Qishan Zhu, Yuxin Wen, and Hongming Piao. Mtcut: A multi-task framework for ranked list truncation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, WSDM '22. Association for Computing Machinery, 2022.

[Wu *et al.*, 2021] Chen Wu, Ruqing Zhang, Jiafeng Guo, Yixing Fan, Yanyan Lan, and Xueqi Cheng. Learning to truncate ranked lists for information retrieval. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[Xi *et al.*, 2023] Yunjia Xi, Jianghao Lin, Weiwen Liu, Xinyi Dai, Weinan Zhang, Rui Zhang, Ruiming Tang, and Yong Yu. A bird's-eye view of reranking: from list level to page level. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 1075–1083, 2023.

[Yang *et al.*, 2012] Xiwang Yang, Harald Steck, Yang Guo, and Yong Liu. On top-k recommendation using social networks. In *Proceedings of the sixth ACM conference on Recommender systems*, 2012.

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

[Zhao *et al.*, 2023] Zihuai Zhao, Wenqi Fan, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Zhen Wen, Fei Wang, Xiangyu Zhao, Jiliang Tang, et al. Recommender systems in the era of large language models (llms). *arXiv preprint arXiv:2307.02046*, 2023.

[Zhu *et al.*, 2024] Haorui Zhu, Fei Xiong, Hongshu Chen, Xi Xiong, and Liang Wang. Incorporating a triple graph neural network with multiple implicit feedback for social recommendation. *ACM Transactions on the Web*, 2024.

[Zou and Liu, 2023] Xin Zou and Weiwei Liu. Generalization bounds for adversarial contrastive learning, 2023.