From Noise to Order: Learning to Rank via Denoising Diffusion

Anonymous Author(s)

Abstract

In information retrieval (IR), learning-to-rank (LTR) methods have traditionally limited themselves to discriminative machine learning approaches that model the probability of the document being relevant to the query given some feature representation of the query-document pair. In this work, we propose an alternative denoising diffusion-based deep generative approach to LTR that instead models the full joint distribution over feature vectors and relevance labels. While in the discriminative setting, an over-parameterized ranking model may find different ways to fit the training data, we hypothesize that candidate solutions that can explain the full data distribution under the generative setting are better equipped to estimate relevance. With this motivation, we propose DiffusionRank that extends TabDiff, an existing denoising diffusion-based generative model for tabular datasets, to create generative equivalents of classical discriminative pointwise and pairwise LTR objectives. We conduct thorough empirical evaluation on four standard LTR datasets to demonstrate improvements from DiffusionRank models over their discriminative counterparts. Our work points to a rich space for future research exploration on how we can leverage ongoing advancements in deep generative modeling approaches, such as diffusion, for learning-to-rank in IR.

CCS Concepts

• **Information systems** → **Learning to rank**; • **Computing methodologies** → **Learning to rank**; **Neural networks**.

1 Introduction

In information retrieval (IR), Learning-to-Rank (LTR) [35] is the task of constructing a model to estimate the relevance of an item (*e.g.*, a document) with respect to an information need (*e.g.*, expressed as a query by the user) so that the IR system can sort the items by their estimated relevance scores for presentation to the user. These ranking models are typically trained on labeled datasets containing a set of query-document pairs and corresponding ground-truth relevance labels. Traditionally, the ranking model is trained discriminatively to predict a real-valued score given the feature-vector. Various objectives for training ranking models have been explored for LTR that Liu et al. [35] broadly categorize under: (i) pointwise, (ii) pairwise, and (iii) listwise loss functions. Under these three categories, different loss functions and machine learning (ML) approaches—including support vector machines [66], neural networks [5], and boosted decision trees [60]—have been explored. However, these explorations have historically been limited to discriminative ML approaches, where the ranker models the probability of the relevance label conditioned on the feature vector.

In this work, we propose an alternative deep generative approach to LTR that instead models the joint distribution over feature vectors and relevance labels. Deep generative LTR opens up new avenues to explore how we can leverage recent advancements in deep generative modeling approaches, such as diffusion [20, 51, 52], for ranking in IR. Unlike discriminative training, where the model is trained only to predict relevance labels conditioned on the features, generative training requires the model to learn the full underlying data distribution, including the conditional distribution of subsets of features given other features (and optionally the label). In the discriminative setting, an over-parameterized ranking model may find different ways to fit to the training data. In the presence of a choice between these alternative solutions, we hypothesize that the solution that fits the full joint distribution under the generative setting leads to more accurate relevance estimation.

Generative modeling of different data modalities—*incl.* text [39], images [30], video [61], speech [10], and tabular data [56]—have recently demonstrated significant improvements in downstream applications. Generative modeling of tabular datasets has found applications in missing value imputation [68], training data augmentation [13], and data privacy protection [1, 19]. Several deep generative models have been proposed for modeling tabular datasets with autoregressive models [2], Variational Autoencoders (VAEs) [34], Generative Adversarial Networks (GANs) [62], and diffusion models [24–27, 49, 67, 68]. In this work, we extend generative approaches to modeling tabular data to LTR datasets to jointly model numerical LTR features and categorical relevance labels. Specifically, we build on TabDiff's [49] continuous-time diffusion process to propose a family of diffusion-based LTR models that we call *DiffusionRank*.

Tabular data presents unique challenges for generative modeling because they comprise heterogeneous columns with varied distributions and data types—*e.g.*, in the LTR datasets, numerical features are continuous while categorical relevance labels are discrete. TabDiff [49] employs a joint diffusion process to simultaneously perturb continuous and discrete data, and learns a joint denoising model for all modalities. We extend TabDiff to LTR datasets where we consider the ranking features as continuous and the relevance label—which maybe either pointwise or a preference label between a pair of documents for a query—as discrete data. This formulation, that we refer to as DiffusionRank, allows us to flexibly extend TabDiff's mixed-type diffusion process to develop generative equivalents to traditional discriminative LTR objectives. Furthermore, TabDiff employs a masked diffusion process for categorical data that involves a single-step denoising (*i.e.*, *unmasking*) at inference time. This has important efficiency implications for DiffusionRank as we need to run the inference of the model only once to estimate the query-document relevance score, identical to their discriminative LTR counterparts. Unlike TabDiff, which employs Transformers for the denoising model, we parameterize DiffusionRank with a feed-forward network with a negligible increase in learnable parameters over an equivalent discriminative model, further ensuring that inference time costs are comparable for generative and discriminative LTR. Furthermore, DiffusionRank does not mandate any specific base model and can flexibly incorporate new advances in neural architecture design.

To summarize our contributions: We propose DiffusionRank, a diffusion-based deep generative LTR model, that scouts a potential path forward to leverage the emerging advancements in diffusion

modeling for the LTR task. Our contributions include the formalization of DiffusionRank and extensive empirical evaluation on four standard LTR datasets to demonstrate improved accuracy of relevance estimation from generative LTR approaches. In this preliminary work, we focus on the pointwise and pairwise settings to rigorously test the hypothesis that generative LTR approaches improve over their discriminative counterparts, and we expect future work to extend these methods to listwise and other advanced LTR settings to achieve state-of-the-art performances. We believe that this work lays a foundation towards developing more expansive generative research agendas in IR.

Next, we formally introduce some of the foundational concepts in LTR and diffusion processes that we build on in our current work, and discuss related literature, in Section 2. Then, we describe the DiffusionRank model in Section 3. In Section 4, we describe our experimental methodology and evaluation protocols, before presenting our results and analysis in Section 5. Finally, we conclude in Section 6 with a discussion on potential new research directions for future work on generative approaches to ranking in IR.

2 Preliminaries and Related Work

In order to situate our work within the broader landscape of generative approaches to learning-to-rank, we first establish the necessary background for our proposed approach; this section therefore both introduces the key preliminaries we build on and clarifies how our method relates to—and differs from—prior work.

2.1 Learning-to-Rank

Formally, let a rankable document d in context of the query q be represented by a feature vector $\vec{x}_{q,d} \in \mathbb{R}^n$, and its ground-truth relevance label with respect to q be represented as $y_{q,d} \in \{0, 1, \ldots, R-1\}$, assuming a R-point labeling scheme. The feature vector representing a query-document pair may comprise of manually-designed numeric features—*e.g.*, BM25 [48] and PageRank [3]—or, one-hot encoding of query and document tokens, in representation-learning ranking models [41]. The labeling scheme, in its simplified form, may use binary relevance assessments, *i.e.*, $R = 2$ and $y_{q,d} \in \{0, 1\}$, categorizing each document to be either *non-relevant* or *relevant* to the query. Or alternatively, it can employ graded-relevance assessments—*e.g.*, the 5-point labeling scheme from Bing [38] that categorizes each document's relevance to the query as one of {Perfect, Excellent, Good, Fair, Bad}. The ranking model $f : \vec{x}_{q,d} \rightarrow \mathbb{R}$ takes the feature-vector $\vec{x}_{q,d}$ as input and predicts a real-valued score $s_{q,d} \in \mathbb{R}$ commensurate with its estimate of the document's relevance to the query. The ranking model may be trained with pointwise, pairwise, or listwise objectives [35, 41].

2.1.1 Pointwise objectives. In pointwise training, the ranking model is optimized to predict the ground-truth label for a query-document pair. If the relevance label is categorical, then its prediction can be treated as a multiclass classification problem [31]. The model under this setting estimates the probability distribution over label categories and can be trained using the cross-entropy (CE) loss.

$$\mathcal{L}_{\text{pointwise-CE}} = -\log\big(p(y_{q,d}|\vec{x}_{q,d})\big) \tag{1}$$

If the relevance labels are binary, then the predicted probability of the document being relevant to the query can be directly used for ranking. However, for graded-relevance labeling schemes, the probability distribution over the label categories must be aggregated to generate a single ranking score.

Pointwise ranking models can also be trained using regression objectives, such as the squared loss, where $y_{q,d}$ and $s_{q,d}$ are either represented as absolute values [9] or as one-hot encodings [15].

$$\mathcal{L}_{\text{pointwise-squared}} = \|y_{q,d} - s_{q,d}\|^2 \tag{2}$$

2.1.2 Pairwise objectives. In contrast, pairwise training [5, 7, 14, 18] optimizes the ranking model to minimize the number of preference errors between pairs of documents for the same query (*i.e.*, when $y_{q,d_i} > y_{q,d_j}$ but $s_{q,d_i} < s_{q,d_j}$) in the training data. Pairwise ranking losses generally take the following form‘[7].

$$\mathcal{L}_{\text{pairwise}} = \phi(s_{q,d_i} - s_{q,d_j}) \tag{3}$$

Where, ϕ can be the Hinge function [18], the exponential function [14], or the logistic function [5]. When the logistic function is used, the resulting loss function is referred to as RankNet [5]. If we sort the pair of documents, such that $y_{q,d_i} > y_{q,d_j}$, then

$$\mathcal{L}_{\text{RankNet}} = -\log\big(1 + e^{-(s_{q,d_i} - s_{q,d_j})/\tau}\big) \tag{4}$$

In Section 3, we define diffusion-based generative LTR objectives corresponding to the discriminative losses $\mathcal{L}_{\text{pointwise-CE}}$ and $\mathcal{L}_{\text{RankNet}}$.

Specialized LTR. There is also an extensive body of work specializing LTR to different settings. This includes LTR trained with different sources of supervision (*e.g.*, online interactions with users [21] and biased feedback data [23]), LTR with specific attributes (*e.g.*, stochasticity [4, 12], uncertainty-awareness [8], and interpretability [70]), and LTR with additional constraints (*e.g.*, diversity [45] and fairness [12, 50]). These topics are out of scope for our current discussion but the application of generative LTR to these specialized settings may offer interesting future directions for research.

2.2 Diffusion modeling

2.2.1 Denoising Diffusion Probabilistic Models. Denoising Diffusion Probabilistic Models (DDPMs) [20, 51] are a class of latent variable generative models inspired by non-equilibrium thermodynamics. Unlike GANs that learn an implicit generative function, diffusion models are likelihood-based and learn to reverse a gradual noise-adding process. The framework consists of two processes: a forward diffusion process and a reverse denoising process.

Forward Process. The forward process is a fixed Markov chain that gradually adds Gaussian noise to the data $\vec{x}_0 \sim q(\vec{x}_0)$ over T steps. At each step t, the transition is defined as:

$$q(\vec{x}_t|\vec{x}_{t-1}) = \mathcal{N}(\vec{x}_t; \sqrt{1-\beta_t}\vec{x}_{t-1}, \beta_t\mathbf{I}) \tag{5}$$

where $\{\beta_t\}_{t=1}^T$ is a predefined variance schedule. Crucially, this process allows us to sample $\vec{x}_t$ at any arbitrary time step t directly from $\vec{x}_0$ in closed form:

$$q(\vec{x}_t|\vec{x}_0) = \mathcal{N}(\vec{x}_t; \sqrt{\bar{\alpha}_t}\vec{x}_0, (1-\bar{\alpha}_t)\mathbf{I}) \tag{6}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$. As $T \rightarrow \infty$, the data $\vec{x}_0$ is transformed into pure Gaussian noise $\vec{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

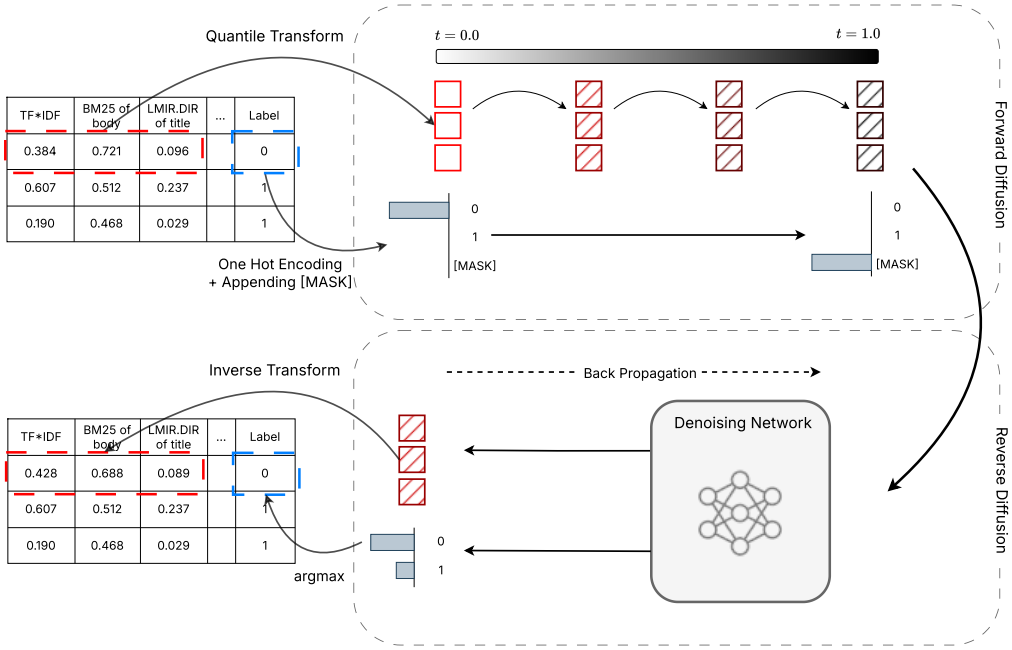

Figure 1: Overview of DiffusionRank (pointwise). We model learning-to-rank as denoising diffusion process over tabular feature-label tuples: a forward process progressively corrupts the input (Gaussian noise for numerical features and masking for categorical variables, including relevance labels), and a learned reverse process denoises to recover clean samples.

Reverse Process. The goal of the generative model is to reverse this process, starting from pure noise $\vec{x}_T$ and sequentially denoising it to recover a sample from the data distribution. Since the true posterior $q(\vec{x}_{t-1}|\vec{x}_t)$ is intractable, DDPMs learn a parameterized approximation $p_\theta(\vec{x}_{t-1}|\vec{x}_t)$:

$$p_\theta(\vec{x}_{t-1}|\vec{x}_t) = \mathcal{N}(\vec{x}_{t-1}; \boldsymbol{\mu}_\theta(\vec{x}_t, t), \Sigma_\theta(\vec{x}_t, t)) \tag{7}$$

To train the model, Ho et al. [20] simplified the variational lower bound on the negative log-likelihood. Instead of predicting the mean $\boldsymbol{\mu}_\theta$ directly, the model $\boldsymbol{\epsilon}_\theta(\vec{x}_t, t)$ is typically trained to predict the noise $\boldsymbol{\epsilon}$ that was added to $\vec{x}_0$ to generate $\vec{x}_t$. The simplified training objective is a mean-squared error loss:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{t,\vec{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\vec{x}_t, t)\|^2\right] \tag{8}$$

While originally formulated for continuous data using Gaussian noise, this framework has recently been extended to continuous-time stochastic differential equations (SDEs) [53] and other data types [63]. In this work, we leverage these advancements to model the joint distribution of ranking features and relevance labels.

2.2.2 Modeling Tabular Data With TabDiff. TabDiff [49] operates on mixed-tabular datasets containing numerical and categorical features. Let $\vec{z} = [\vec{x}, \vec{y}]$ represent a data sample consisting of a concatenation of vectors of C_{num} continuous numerical features $\vec{x}$ and C_{cat} categorical features $\vec{y}$. The vector of categorical features $\vec{y}$ is in turn are a concatenation of the C_{cat} individual categorical features $\vec{y}_i \in \{0,1\}^{(c_i+1)}$ which are one-hot encoded, with c_i as the number of categories in feature $\vec{y}_i$. The one-hot encoding of $\vec{y}_i$ includes an additional dimension to indicate the [MASK] state for features during the masked diffusion process.

TabDiff, like other diffusion models, is a likelihood-based generative model that learns the data distribution using forward and backward Markov processes. TabDiff gradually corrupts the data in the forward process and learns a denoising model to recover the original data in the reverse process. Let $\{\vec{z}_t : t \sim [0,1]\}$ denote a sequence of data in the diffusion process where $t \in [0,1]$ is a continuous time variable, such that $\vec{z}_0 \sim p_0$ is a real data sample and $\vec{z}_1 \sim p_1$ is pure noise from a prior distribution. We can represent the forward diffusion process as follows:

$$q(\vec{z}_t|\vec{z}_0) = q(\vec{x}_t|\vec{x}_0, \sigma_{\text{num}}) \cdot q(\vec{y}_t|\vec{y}_0, \sigma_{\text{cat}}) \tag{9}$$

Where, σ_{num} and σ_{cat} are the diffusion schedules for numerical and categorical features, respectively. The true reverse process can consequently be represented as:

$$q(\vec{z}_s|\vec{z}_t, \vec{z}_0) = q(\vec{x}_s|\vec{z}_t, \vec{z}_0) \cdot q(\vec{y}_s|\vec{z}_t, \vec{z}_0) \tag{10}$$

Where, s and t represent two arbitrary time steps such that $0 < s < t < 1$. TabDiff learns a denoising model $p_\theta(\vec{z}_s|\vec{z}_t)$ to approximate the true posterior $q(\vec{z}_s|\vec{z}_t, \vec{z}_0)$. This denoising model is learnt by minimizing the following loss:

$$\mathcal{L}_{\text{TabDiff}} = \lambda_{\text{num}} \cdot \mathcal{L}_{\text{num}} + \lambda_{\text{cat}} \cdot \mathcal{L}_{\text{cat}} \tag{11}$$

Where $\mathcal{L}_{\text{num}}$ and $\mathcal{L}_{\text{cat}}$ represent the loss components corresponding to the numerical and categorical features, respectively, and λ_{num} and λ_{cat} are two weight terms. Let, $\vec{\chi}$ and $\vec{\psi}$ denote the numerical and categorical part of the denoising model's output, respectively. We next describe how we compute $\mathcal{L}_{\text{num}}$ and $\mathcal{L}_{\text{cat}}$ with respect to the denoising model outputs $\vec{\chi}$ and $\vec{\psi}$.

Gaussian diffusion for numerical features. TabDiff's forward diffusion process gradually corrupts the numerical features by adding sampled noise to them, as represented below:

$$\vec{x}_t = \vec{x}_0 + \sigma_{\text{num}}(t) \cdot \vec{\epsilon}, \quad \vec{\epsilon} \sim \mathcal{N}(\vec{0}, \vec{\mathrm{I}}_{C_{\text{num}}}) \tag{12}$$

The numerical part of the denoising model then tries to predict the noise that was added, and trains by minimizing the following loss:

$$\mathcal{L}_{\text{num}} = \mathbb{E}_{t\sim U[0,1]}\mathbb{E}_{(\vec{z}_t,\vec{z}_0)\sim q(\vec{z}_t,\vec{z}_0)} \|\vec{\chi}(t) - \vec{\epsilon}\|_2^2 \tag{13}$$

Masked diffusion for categorical features. For categorical features, the forward diffusion process is defined as a masking process, represented as follows:

$$q(\vec{x}_t|\vec{x}_0) = \text{Cat}\big(\vec{x}_t; \alpha_t \cdot \vec{x}_0 + (1-\alpha_t)\text{m}\big) \tag{14}$$

Where, $\text{Cat}(;\pi)$ is a categorical distribution over the classes with probabilities given by π, and $\alpha_t \in [0,1]$ is a strictly-decreasing function of t, with the additional constraints that $\alpha_0 \approx 1$ and $\alpha_1 \approx 0$. At each diffusion step, the feature is corrupted by being changed to the [MASK] state with a probability of $(1-\alpha_t)$ and then remains as such till $t = 1$. At time $t = 0$, all categorical features are unmasked; and at $t = 1$, all of them are masked.

In the reverse denoising process, the model aims to recover the original feature values from the masked state. When a feature is masked in the input, the model predicts it conditioned on the remaining noisy features at time t. During training, the loss is computed only on predictions corresponding to inputs in the masked state, ensuring that the model learns to unmask corrupted features rather than trivially copying unmasked ones. The categorical part of the denoising model is trained by minimizing the following loss, where α'_t is the first order derivative of α_t:

$$\mathcal{L}_{\text{cat}} = \mathbb{E}_{t\sim U[0,1]}\mathbb{E}_{(\vec{z}_t,\vec{z}_0)\sim q(\vec{z}_t,\vec{z}_0)}\left[\frac{\alpha'_t}{1-\alpha_t} \cdot \log\langle\vec{\psi}(t), \vec{y}_0\rangle\right] \tag{15}$$

We point the reader to Shi et al. [49] for further details about TabDiff.

2.3 Alternative Approach(es) to Generative LTR

While the LTR research community has focused almost exclusively on discriminative ML approaches, IRGAN Wang et al. [57] and subsequent follow up works—*e.g.*, [11, 22, 28, 29, 36, 42, 64, 65]—constitute one strand of IR research that has explored Generative Adversarial Networks (GANs) [17] for ranking. However, that line of work has focused more on the *adversarial* aspects of GAN, rather than its *generative* aspects. The IRGAN formulation involves two models: (a) a generator model $p_\theta(d|q,y)$ which identifies non-relevant documents for a query from a candidate pool that closely resemble relevant documents in some feature space, and (b) a discriminator model that tries to discriminate between the relevant and the adversarially selected non-relevant document. Unlike our work, IRGAN does not learn a joint distribution over query-document features $\vec{x}_{q,d}$ and the relevance label $y_{q,d}$. Therefore, further comparisons with IRGAN are out of scope for our work.

The other strand of related work includes the use of generative large language models (LLMs) for ranking [16, 44, 54, 58, 69]. These approaches prompt LLMs trained with generative modeling objectives to estimate query-document relevance. However, these approaches do not focus on *learning* to rank, and hence are also out of scope for our current work.

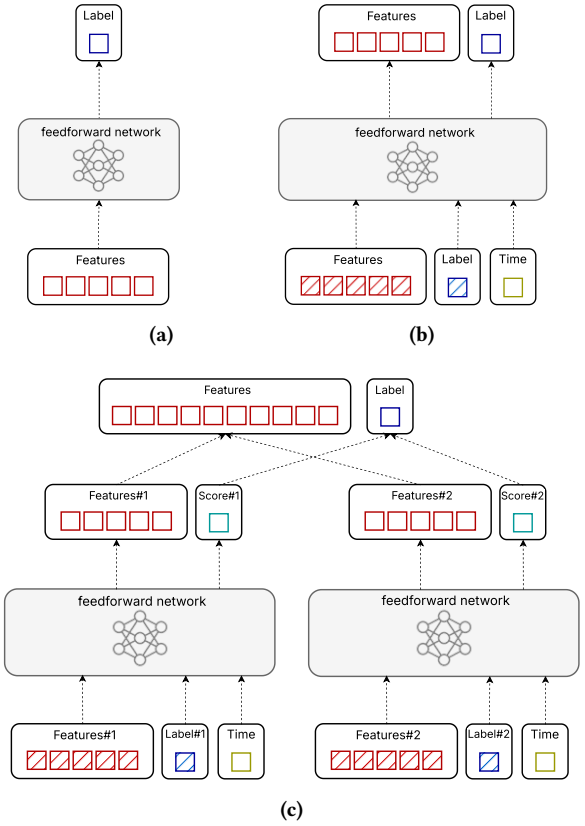

Figure 2: Architectural differences between discriminative and DiffusionRank models. (a) Standard discriminative LTR model, which takes only query-document features as input and predicts a relevance score or label. (b) Pointwise DiffusionRank, where the model additionally conditions on the (possibly masked) relevance label and diffusion time step, and jointly predicts the relevance label and the noise added to features. (c) Pairwise DiffusionRank, which applies the pointwise denoising model independently to a document pair with tied label masking, producing scores for both documents while learning from noisy feature representations.

3 DiffusionRank

DiffusionRank extends TabDiff to LTR datasets by treating the ranking features and the relevance labels (both pointwise or pairwise) as numerical and categorical tabular data, respectively. Figure 1 shows the forward and reverse diffusion processes of our training pipeline. Like TabDiff, we employ Gaussian diffusion over the LTR features and masked diffusion for predicting the relevance label.

We parameterize the diffusion process with a feedforward network, an architecture choice that is common in the LTR settings. DiffusionRank can flexibly incorporate any alternative neural architectures as long as they can be extended to: (i) Accept the label, which may be masked or unmasked, and the time step t of the diffusion process as additional inputs, and (ii) predict the noise over the features jointly with the relevance labels. Figure 2 sketches out

these necessary modifications to the base model from the discriminative LTR setting. Next, we elaborate on how we instantiate this framework for pointwise and pairwise LTR settings.

3.1 Pointwise DiffusionRank

Pointwise DiffusionRank is a straightforward application of the framework just described. The LTR model $f : \vec{x}_{q,d} \rightarrow \mathbb{R}$ from the discriminative setting is modified to operate as a denoising model $\bar{f} : [\vec{x}_{q,d}, y_{q,d}, t] \rightarrow [\mathbb{R}^{C_{\text{num}}}, \mathbb{R}^{C_{\text{cat}}+1}]$ as shown in Figure 2b, where C_{num} is the number of LTR features in the dataset and $C_{\text{cat}} = k$ for a k-point relevance labeling scheme. The relevance label, as both input to and output of the model, is one-hot encoded with an extra [MASK] dimension. As part of the diffusion process, the model is trained to predict the relevance label, given (i) the features corrupted by adding noise, (ii) time step t, and (iii) the label as masked in the input. This is equivalent to training a discriminative model with the $\mathcal{L}_{\text{pointwise-CE}}$ loss, where the features have been corrupted and two additional features (the masked label and the time step t) are added as input. Furthermore, the model also predicts the estimated noise with respect to the feature vector which encourages the model to learn the joint distribution of features and labels in the dataset.

At inference time, we provide as input (i) the uncorrupted features, (ii) time step $t = 0$, and (iii) [MASK] as the label; and use the model to directly predict the relevance of the query-document pair. The part of the final layer of the model that predicts the feature noise can be safely ignored at inference time to keep the runtime cost almost exactly similar as that of its discriminative counterpart. To further ensure comparable inference-time compute and time costs between the generative and discriminative settings, we do not use the backward stochastic sampler and the guidance classifier employed by Shi et al. [49] in their original work.

3.2 Pairwise DiffusionRank

Unlike in the pointwise setting, in pairwise training we have two feature vectors $\vec{x}'_{q,d_i}$ and $\vec{x}'_{q,d_j}$ corresponding to the pair of documents for the same query, and a preference label y denoting whether document d_i is more relevant than d_j, or not. From the TabDiff perspective, we consider $\vec{x}_t = [\vec{x}'_{q,d_i,t}, \vec{x}'_{q,d_j,t}]$ as the concatenation of the two feature vectors; and in the Gaussian diffusion process, the noise added to the two feature vectors are sampled independently.

In spite of the pairwise setting, we employ a pointwise model $\bar{f} : [\vec{x}_{q,d}, y_{q,d}, t] \rightarrow [\mathbb{R}^{C_{\text{num}}}, \mathbb{R}]$ for the denoising. This is similar to standard discriminative pairwise settings, where the loss is pairwise (*i.e.*, considers $\langle q, d_i, d_j \rangle$) but the models themselves are typically pointwise (*i.e.*, they take $\langle q, d \rangle$ as input). This pointwise model is applied to $\vec{x}'_{q,d_i}$ and $\vec{x}'_{q,d_j}$ independently in parallel. The labels for the documents are tied-masked—*i.e.*, they are either both masked or unmasked at the same time. When unmasking the label as part of the reverse diffusion process, the preference label prediction $\vec{\psi} = [s_{q,d_i}, s_{q,d_j}]$ is a concatenation of the scores s_{q,d_i} and s_{q,d_j}. This is similar to training the model with $\mathcal{L}_{\text{RankNet}}$ with the addition of noise to the features and the additional Gaussian diffusion loss as a component of the overall optimization objective.

In this work, we designed diffusion-based counterparts to $\mathcal{L}_{\text{pointwise-CE}}$ and $\mathcal{L}_{\text{RankNet}}$ to demonstrate the generality of our method and its relevance to different LTR approaches. We leave similar adaptation of other discriminative LTR objectives, including listwise losses, into the DiffusionRank framework for future work.

Table 1: Summary statistics of the LTR datasets used in our evaluation. All reported values correspond to Fold 1

	WEB30K	WEB10K	Istella-S	MQ2007
Queries (Train)	18,919	6,000	19,245	1,017
Queries (Val)	6,306	2,000	7,211	339
Queries (Test)	6,306	2,000	6,562	336
Data Points (Total)	3,771,125	1,200,192	3,408,630	69,623
Per Query (avg.)	119.60	120.01	103.23	41.14
Features (F)	136	136	220	46
Relevance Labels (R)	5	5	5	3

4 Experiments

4.1 Data

We evaluate our approach on four widely used Learning to Rank datasets: MQ2007 partition of LETOR 4.0, MSLR-WEB10K, MSLR-WEB30K [43], and Istella-S [37]. Both LETOR 4.0 and MSLR-WEBs are constructed from real Bing search logs, with MSLR-WEB10K being approximately fourteen times larger than LETOR 4.0, whereas Istella-S is derived from the Istella search engine and provides a large-scale dataset with over 3.4 million query-document pairs.

Each query-document pair in LETOR 4.0 is represented by 46 numerical features, and relevance labels are graded from 0 to 2. In MSLR-WEBs, each pair has 136 features, and relevance labels range from 0 (irrelevant) to 4 (perfectly relevant). For Istella-S, each pair is represented by 220 features, and relevance labels follow the same 0 to 4 grading scale. Table 1 summarizes dataset statistics.

Our pointwise formulation assumes binary relevance labels in the training data to simplify the joint distribution modeled by the diffusion process, allowing the model to efficiently capture the fundamental distinction between relevant and non-relevant documents. So, for pointwise training, we binarize the relevance labels in our datasets: for LETOR 4.0, $0 \rightarrow 0$ and $1,2 \rightarrow 1$; for MSLR-WEBs and Istella-S, $0,1 \rightarrow 0$ and $2,3,4 \rightarrow 1$. The original labels were retained for evaluation. All features were transformed using a Quantile Transformer. The same transformation was also applied when training the baseline models to ensure a fair comparison.

For experiments analyzing the effect of training data size, we randomly sampled subsets of training data and used the same samples across all runs. This ensures all models were trained on exactly the same data, while the validation and test remained unchanged.

4.2 Models

To ensure a principled and fair comparison between generative and discriminative approaches, we adopt the same base neural architecture across all models. This design choice allows us to isolate the effect of the learning paradigm itself without confounding factors arising from architectural differences.

Both discriminative and generative models are implemented using a feedforward network (FFN) backbone with four hidden layers. The hidden layer size is set to 256 for the MQ2007 dataset and 1,024 for MSLR-WEB10K, MSLR-WEB30K, and Istella-S. Each

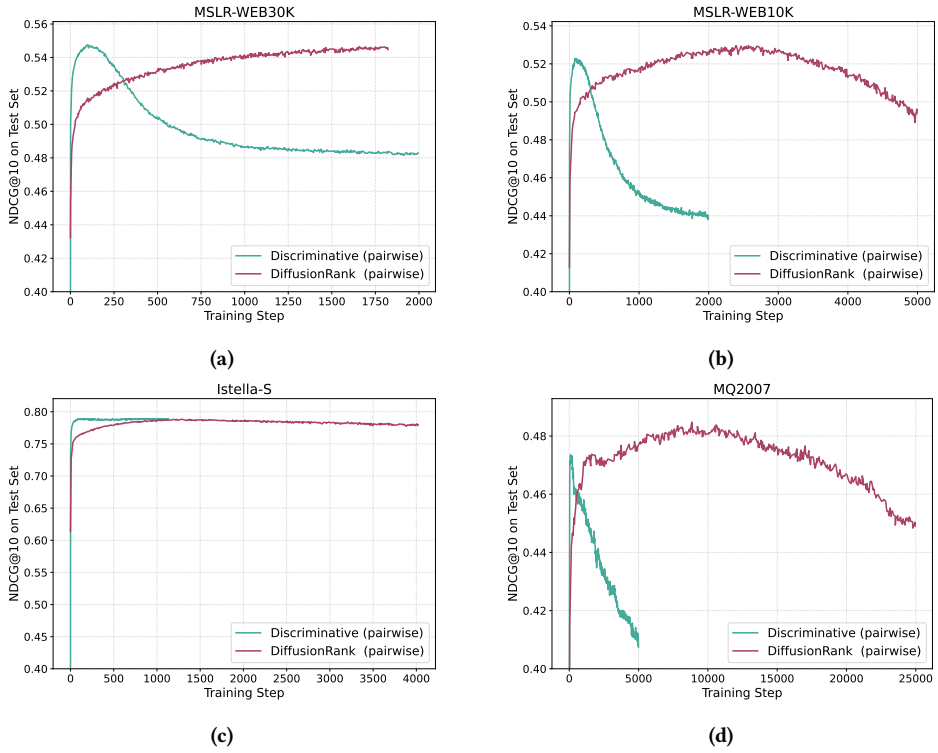

Figure 3: Training dynamics on test data (NDCG@10) for discriminative vs. DiffusionRank models. We plot validation effectiveness over training steps (a) MSLR-WEB30K, (b) MSLR-WEB10K, (c) Istella-S, and (d) MQ2007. Across all datasets except Istella-S, DiffusionRank shows smoother, more stable trajectories and less pronounced degradation at later training stages, consistent with improved robustness to overfitting compared to discriminative baselines.

hidden layer applies SiLU activation followed by Layer Normalization and a dropout rate of 0.1. Detailed hyperparameter tuning configurations are further elaborated in Section 4.3. As previously mentioned in Section 3, in DiffusionRank, the model accepts as input both noisy features and labels and predicts feature noise and the correct label. Figure 2 illustrates these differences.

The XGBoost baseline [6] is trained using 'binary:logistic' objective for pointwise and 'rank:pairwise' for pairwise tasks. Maximum tree depth is set to 6 for MQ2007 and 32 for other datasets, matching the scale of the neural models.

4.3 Training

Neural models are trained using AdamW optimizer. DiffusionRank employs a continuous-time diffusion process with 50 steps for numeric and categorical features. For the categorical loss, we set $\lambda_{\text{cat}} = 1$ and apply an annealing scheduler on λ_{num} to gradually reduce its importance during training, following TabDiff settings [49].

The diffusion process is further parameterized with noise schedulers: 'power_mean' for numeric features and 'log_linear' for categorical features. At each step, noise is added to the features and labels, with the diffusion time variable t sampled from a uniform distribution. These settings allow the model to balance feature and label denoising effectively throughout the diffusion process.

Hyperparameter tuning was performed once using random search for both the discriminative and generative models. The search space encompassed the following parameters: data normalization method (comparing the original, unnormalized data against data transformed via a Quantile Transformer), learning rate (sampled within the range of 4e-1 to 5e-8), number of hidden layers (ranging from 2 to 4), hidden layer size (64, 128, 256, 512, 1024, 2048), activation function (SiLU, ReLU, or Sigmoid), dropout rate (0.0, 0.1, 0.2, 0.3, 0.4, 0.5), and loss-weight schedules (for the generative models). Surprisingly, we observed that the majority of the optimal hyperparameters were identical across both modeling approaches, demonstrating consistent peak performance regardless of the chosen architecture. Given this consistency, we utilized the best-performing discriminative parameters for the generative models as well. Ultimately, the best model was selected based on the highest NDCG@10 score achieved on the validation set. All models and baselines were trained on a single NVIDIA RTX A6000 GPU with 48GB of memory. Note that we built our framework on top of TabDiff's code[1]. To support reproducibility and facilitate future research, we made all code publicly available on GitHub[2].

4.4 Evaluation

We evaluate all models using standard ranking effectiveness metrics commonly adopted in Learning-to-Rank research. Specifically, we report Normalized Discounted Cumulative Gain (NDCG) and Mean

[1]https://github.com/MinkaiXu/TabDiff
[2]https://anonymous.4open.science/r/DiffusionRank-1E51/

Table 2: Results of the pointwise approaches on MSLR-WEB30K, MSLR-WEB10K, Istella-S, and MQ2007. We compare discriminative baselines and DiffusionRank under pointwise training for multiple training data fractions K (smaller K corresponds to less training data). Performance for XGBoost is included solely for comparison with traditional models and is not part of our core hypothesis. The highest metric value for each K is bolded.

	MSLR-WEB30K		MSLR-WEB10K		Istella-S		MQ2007	
	NDCG@10	MAP@10	NDCG@10	MAP@10	NDCG@10	MAP@10	NDCG@10	MAP@10
K=1.0								
XGBoost	0.5187	0.7613	0.4866	0.7374	**0.7851**	**0.9346**	0.4397	0.5221
Discriminative	0.5401	0.7769	0.5123	0.7568	0.7742	0.9299	0.4640	**0.5403**
DiffusionRank	**0.5424**	**0.7798**	**0.5202**	**0.7676**	0.7828	0.9317	**0.4751**	0.5377
K=2^{-2}								
XGBoost	0.5050	0.7534	0.4761	0.7321	**0.7704**	**0.9271**	0.4076	0.4794
Discriminative	0.5244	0.7663	0.5015	0.7508	0.7595	0.9179	0.4468	0.5345
DiffusionRank	**0.5345**	**0.7767**	**0.5066**	**0.7563**	0.7696	0.9258	**0.4614**	**0.5377**
K=2^{-4}								
XGBoost	0.4779	0.7350	0.4562	0.7137	0.7535	**0.9184**	0.4134	0.4852
Discriminative	0.5044	0.7567	0.4791	0.7370	0.7444	0.9075	0.4458	**0.5226**
DiffusionRank	**0.5206**	**0.7688**	**0.4816**	**0.7391**	**0.7541**	0.9132	**0.4475**	0.5183
K=2^{-6}								
XGBoost	0.4711	0.7320	0.4390	0.7080	0.7332	**0.9062**	0.3618	0.4487
Discriminative	0.4865	0.7460	**0.4610**	**0.7252**	0.7237	0.8954	0.3908	0.4813
DiffusionRank	**0.5050**	**0.7585**	0.4557	0.7210	**0.7371**	0.9004	**0.3979**	**0.4840**
K=2^{-8}								
XGBoost	0.4525	0.7189	0.4024	0.6700	0.7031	0.8778	0.3661	0.4459
Discriminative	0.4709	0.7388	0.4227	**0.6877**	0.6984	0.8769	0.3655	0.4495
DiffusionRank	**0.4891**	**0.7527**	**0.4253**	0.6839	**0.7133**	**0.8796**	**0.3731**	**0.4626**

Average Precision (MAP) at a cutoff of 10, with all scores averaged over queries in each dataset. While pointwise models are trained using binarized relevance labels, all evaluations are conducted using the original graded relevance judgments provided by the datasets to ensure a fair and standard assessment of ranking quality across all settings. To assess statistical significance between models, we apply a paired t-test with a significance threshold of $p < 0.05$.

5 Results

In this section, we first present the overall effectiveness of DiffusionRank compared to standard discriminative learning-to-rank baselines. We then drill down into two factors that help explain when and why generative training is beneficial: (i) the impact of training data size on DiffusionRank's relative advantage, and (ii) an ablation that perturbs input features during discriminative training to test whether DiffusionRank's gains could be attributed to robustness induced by noisy inputs rather than to modeling the joint feature-label distribution.

5.1 Generative *vs.* Discriminative LTR

We report the performance of different methods across all datasets in terms of NDCG@10 and MAP@10 in Table 2 and Table 3. Different sections of the tables illustrate results for multiple values of K, reflecting different fractions of the training data used to train the model. As shown in these tables, when training with full data (i.e., K=1), DiffusionRank consistently outperforms its discriminative counterparts in both pointwise and pairwise formulations on MQ2007, MSLR-WEB10K, and MSLR-WEB30K. On Istella-S, this performance advantage was maintained in the pointwise setting. For the datasets where DiffusionRank succeeded, generative pointwise models achieve higher effectiveness than discriminative pointwise models, and generative pairwise models similarly outperform discriminative pairwise baselines. These improvements are statistically significant based on paired t-tests with $p < 0.05$.

Table 2 and Table 3 also report performance as we progressively reduce the amount of training data (smaller K). As expected, effectiveness generally degrades for all methods as K decreases. This performance lead persists on MSLR-WEB10K, MSLR-WEB30K, and Istella-S, where DiffusionRank typically retains an advantage over the corresponding discriminative baselines across a wide range of data fractions (both pointwise and pairwise), suggesting improved robustness in the low-to-moderate data regime. On the MQ2007, the trends for pairwise approaches are more mixed: DiffusionRank tends to match or improve upon discriminative baselines at moderate fractions (e.g., $K = 2^{-2}$ and $K = 2^{-4}$), but differences become noisier as K becomes very small. Overall, these results indicate that DiffusionRank's gains are most consistent when sufficient training data is available; in this regime, it consistently outperforms its discriminative counterparts.

To better understand these gains, Figure 3 illustrates the training dynamics of discriminative and generative models. We observe a distinct trade-off in training behavior: although the generative approaches exhibit a slower learning curve, they consistently reach a higher performance ceiling in terms of NDCG@10 compared to

Table 3: Results of the pairwise approaches on MSLR-WEB30K, MSLR-WEB10K, Istella-S, and MQ2007. We compare discriminative baselines and DiffusionRank under pairwise training for multiple training data fractions K (smaller K corresponds to less training data). Performance for XGBoost is included solely for comparison with traditional models and is not part of our core hypothesis. The highest metric value for each K is bolded.

	MSLR-WEB30K		MSLR-WEB10K		Istella-S		MQ2007	
	NDCG@10	MAP@10	NDCG@10	MAP@10	NDCG@10	MAP@10	NDCG@10	MAP@10
K=1.0								
XGBoost	0.4355	0.6704	0.4143	0.6504	0.5288	0.7132	0.4005	0.4926
Discriminative	0.5440	0.7844	0.5227	0.7692	**0.7890**	0.9204	0.4582	0.5262
DiffusionRank	**0.5484**	**0.7895**	**0.5288**	**0.7751**	0.7879	**0.9277**	**0.4810**	**0.5512**
K=2^{-2}								
XGBoost	0.4298	0.6677	0.3844	0.6216	0.4541	0.6395	0.4358	0.5271
Discriminative	0.5266	0.7733	0.5044	0.7589	0.7706	0.9127	0.4469	0.5181
DiffusionRank	**0.5362**	**0.7805**	**0.5133**	**0.7613**	**0.7753**	**0.9195**	**0.4623**	**0.5315**
K=2^{-4}								
XGBoost	0.4481	0.6964	0.4098	0.6468	0.4800	0.6870	**0.4210**	**0.4973**
Discriminative	0.4995	0.7591	0.4762	0.7377	0.7401	0.9010	0.4121	0.4892
DiffusionRank	**0.5167**	**0.7705**	**0.4947**	**0.7499**	**0.7489**	**0.9071**	0.4109	0.4840
K=2^{-6}								
XGBoost	0.4134	0.6478	0.3600	0.5902	0.4674	0.6560	**0.3934**	**0.4753**
Discriminative	0.4666	0.7369	0.4265	0.6992	0.7069	0.8710	0.3487	0.4334
DiffusionRank	**0.4889**	**0.7531**	**0.4405**	**0.7133**	0.7069	**0.8715**	0.3464	0.4233
K=2^{-8}								
XGBoost	0.3851	0.6306	0.3462	0.5779	0.5639	0.7613	**0.3815**	**0.4538**
Discriminative	0.4190	0.6994	0.3586	0.6385	0.6020	0.8063	0.3255	0.4114
DiffusionRank	**0.4355**	**0.7084**	**0.3787**	**0.6624**	**0.6205**	**0.8156**	0.3248	0.4097

Table 4: Impact of training models with perturbed data reported on MSLR-WEB30K, MSLR-WEB10K, Istella-S, and MQ2007. We compare discriminative baselines trained on clean features vs. perturbed features (training-time noise injection) and DiffusionRank. The best performance for each configuration is bolded. Italicized values indicate cases where training on perturbed data yielded better results than training on the original data for the discriminative baselines.

	MSLR-WEB30K		MSLR-WEB10K		Istella-S		MQ2007	
	NDCG@10	MAP@10	NDCG@10	MAP@10	NDCG@10	MAP@10	NDCG@10	MAP@10
Pointwise								
Discriminative	0.5400	0.7769	0.5123	0.7568	0.7742	0.9299	0.4640	0.5403
Discriminative perturbed	0.5395	0.7765	*0.5141*	*0.7581*	*0.7757*	0.9289	0.4539	0.5212
DiffusionRank	**0.5424**	**0.7798**	**0.5202**	**0.7676**	**0.7828**	**0.9317**	**0.4751**	**0.5377**
Pairwise								
Discriminative	0.5440	0.7844	0.5227	0.7692	0.7890	0.9204	0.4582	0.5262
Discriminative perturbed	0.5416	0.7822	0.5214	0.7645	0.7872	*0.9220*	*0.4599*	*0.5303*
DiffusionRank	**0.5484**	**0.7895**	**0.5288**	**0.7751**	**0.7879**	**0.9277**	**0.4810**	**0.5512**

their discriminative counterparts. While discriminative models often show widening gaps between training and testing performance, the generative models maintain closer alignment between the two. This increased robustness to overfitting may partially explain the consistent performance improvements observed for DiffusionRank across datasets and training paradigms.

5.2 Effect of training data size

We analyze the impact of training data size on ranking effectiveness in Figure 4 for both pointwise and pairwise approaches. Across datasets, the relative gap between generative and discriminative training is not strictly monotonic and varies by dataset and regime, indicating that the benefits of generative training depend on both data scale and the intrinsic difficulty/noise characteristics of the dataset. On MSLR-WEB10K and MSLR-WEB30K, the separation between the solid (DiffusionRank) and dashed (discriminative) curves becomes increasingly consistent as the training set grows. This pattern suggests that DiffusionRank is able to better exploit additional supervision, potentially because modeling the joint distribution over features and relevance labels benefits from richer coverage of the feature-label space and reduces overfitting as the model sees a wider range of query-document configurations. In other words, in

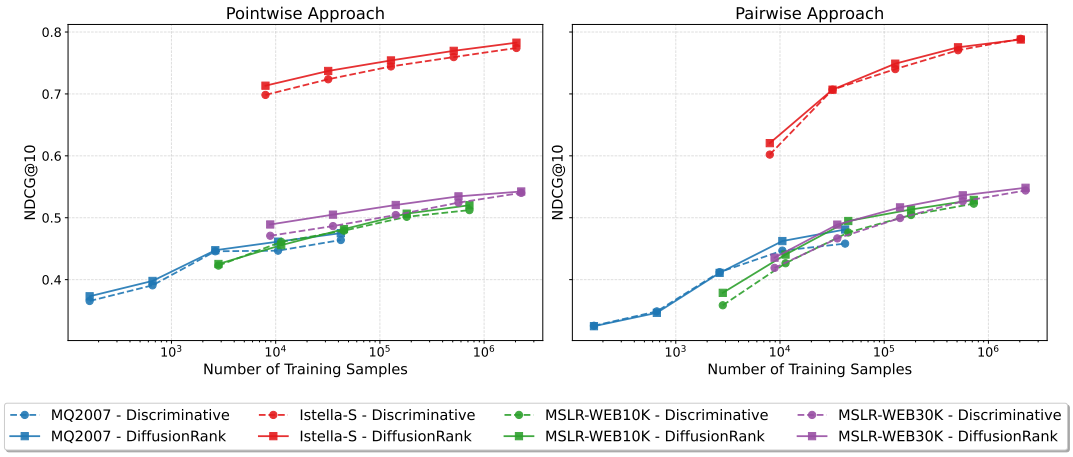

Figure 4: Effect of training set size on ranking effectiveness in terms of NDCG@10. Solid lines denote DiffusionRank and dashed lines denote the corresponding discriminative baseline.

the larger-data regime, the generative objective appears to act as a stronger inductive bias: it encourages solutions that explain the data distribution more globally rather than fitting idiosyncrasies of limited training samples, leading to more reliable gains.

In contrast, on MQ2007, the curves exhibit noticeably higher variance, with occasional crossings in the low to moderate data regime. Moreover, in this regime, the additional modeling burden imposed by the generative objective may not always translate into improved ranking effectiveness, because there may be insufficient data to robustly estimate the joint structure that DiffusionRank is designed to capture. As a result, differences can fluctuate across training sizes, and advantages at one subsample size may not persist.

Overall, while the trends are not fully conclusive for the smaller subsets, the figure supports that DiffusionRank's gains are most stable and pronounced when sufficient training data is available and any claims about data efficiency should be made cautiously.

5.3 Training with noisy features

One could posit that DiffusionRank's improvements are a result of predicting relevance from perturbed inputs that acts as an implicit regularizer, and not because of the joint modeling of features and labels. To demonstrate that this is untrue, we train discriminative pointwise and pairwise models on artificially perturbed (noisy) features and compare them against (i) their standard, non-perturbed counterparts and (ii) DiffusionRank. Table 4 reports NDCG@10 and MAP@10 for each dataset under these conditions. We note that in *perturbed* setting, we inject controlled random noise into the input feature vector of each query–document instance during training, while keeping ground-truth relevance labels unchanged. This perturbation is applied only as a training-time augmentation, aiming to isolate whether the observed gains stem from input noise regularization rather than the generative training objective.

As shown in Table 4, introducing feature perturbations does not consistently improve discriminative learning-to-rank. In fact, discriminative models trained with noisy features perform comparably to their clean-feature counterparts. In contrast, DiffusionRank remains stronger than the discriminative models trained with noisy features, with statistically significant improvements on all four datasets: MQ2007, MSLR-WEB10K, MSLR-WEB30K, and Istella-S ($p < 0.05$, paired t-test). Overall, these results indicate that DiffusionRank's gains cannot be explained solely by a regularization effect from noisy inputs. Instead, they support the conclusion that explicitly learning the joint distribution over features and relevance labels drives the effectiveness of generative LTR in our setting.

6 Conclusion and Future Work

In this work, we demonstrate how diffusion modeling for tabular data presents an exciting new direction for generative LTR. Our primary focus has been to empirically test the hypothesis that modeling the joint distribution of features and labels in generative LTR outperforms their discriminative counterparts in the pointwise and pairwise settings. We hope this motivates future work to translate other discriminative LTR objectives, including listwise losses, to the generative setting.

Research questions for future work also include studying these approaches under large training data regimes, which may be particularly interesting as generating large volumes of training data becomes increasingly cheaper using LLMs-as-Judges [55] and other techniques for synthetic data generation [46, 47]. Generative LTR approaches may also be able to leverage unlabeled LTR datasets consisting only of feature vectors as part of the Gaussian diffusion process, which presents another potentially promising direction for exploration. While our focus in this work has been on manually-designed numerical features, generative LTR may also be extended to representation-learning neural IR models where the inputs are query and document tokens. This would require us to deal with additional modalities as part of our diffusion process; which may benefit from exploring emerging approaches for diffusion over structured datasets like DiSK [25].

While the IR community has not been immune to the excitement around emerging advancements in generative AI [59], the focus has largely been centered on the applications of LLMs, with some notable exceptions [32, 33]. We hope that work, such as ours, will uncover a broader space of opportunities to explore and apply generative modeling algorithms in IR that do not simultaneously carry the baggage of the societal risks associated with LLMs [40].

References

[1] Samuel A Assefa, Danial Dervovic, Mahmoud Mahfouz, Robert E Tillman, Prashant Reddy, and Manuela Veloso. 2020. Generating synthetic data in finance: opportunities, challenges and pitfalls. In *Proceedings of the First ACM International Conference on AI in Finance*. 1–8.

[2] Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. 2022. Language models are realistic tabular data generators. *arXiv preprint arXiv:2210.06280* (2022).

[3] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems* 30, 1-7 (1998), 107–117.

[4] Sebastian Bruch, Shuguang Han, Michael Bendersky, and Marc Najork. 2020. A stochastic treatment of learning to rank scoring functions. In *Proc. WSDM*. 61–69.

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

[6] Tianqi Chen. 2016. XGBoost: A Scalable Tree Boosting System. *Cornell University* (2016).

[7] Wei Chen, Tie-Yan Liu, Yanyan Lan, Zhi-Ming Ma, and Hang Li. 2009. Ranking measures and loss functions in learning to rank. *neurips* 22 (2009).

[8] Daniel Cohen, Bhaskar Mitra, Oleg Lesota, Navid Rekabsaz, and Carsten Eickhoff. 2021. Not All Relevance Scores are Equal: Efficient Uncertainty and Calibration Modeling for Deep Retrieval Models. In *Proc. SIGIR*. ACM.

[9] David Cossock and Tong Zhang. 2006. Subset ranking using regression. In *International conference on computational learning theory*. Springer, 605–619.

[10] Wenqian Cui, Dianzhi Yu, Xiaoqi Jiao, Ziqiao Meng, Guangyan Zhang, Qichao Wang, Steven Y Guo, and Irwin King. 2025. Recent advances in speech language models: A survey. In *acl*. 13943–13970.

[11] Ameet Deshpande and Mitesh M Khapra. 2020. Evaluating a generative adversarial framework for information retrieval. *arXiv preprint arXiv:2010.00722* (2020).

[12] Fernando Diaz, Bhaskar Mitra, Michael D Ekstrand, Asia J Biega, and Ben Carterette. 2020. Evaluating stochastic rankings with expected exposure. In *Proc. CIKM*. 275–284.

[13] Joao Fonseca and Fernando Bacao. 2023. Tabular and latent space synthetic data generation: a literature review. *Journal of Big Data* 10, 1 (2023), 115.

[14] Yoav Freund, Raj Iyer, Robert E Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *Journal of machine learning research* 4, Nov (2003), 933–969.

[15] Norbert Fuhr. 1989. Optimum polynomial retrieval functions based on the probability ranking principle. *ACM Transactions on Information Systems (TOIS)* 7, 3 (1989), 183–204.

[16] Jingtong Gao, Bo Chen, Xiangyu Zhao, Weiwen Liu, Xiangyang Li, Yichao Wang, Wanyu Wang, Huifeng Guo, and Ruiming Tang. 2025. Llm4rerank: Llm-based auto-reranking framework for recommendations. In *WWW*. 228–239.

[17] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *neurips* 27 (2014).

[18] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. 2000. Large margin rank boundaries for ordinal regression. *Advances in Large Margin Classifiers* (2000).

[19] Mikel Hernandez, Gorka Epelde, Ane Alberdi, Rodrigo Cilla, and Debbie Rankin. 2022. Synthetic data generation for tabular health records: A systematic review. *Neurocomputing* 493 (2022), 28–45.

[20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* 33 (2020), 6840–6851.

[21] Katja Hofmann. 2013. *Fast and reliable online learning to rank for information retrieval*. Ph. D. Dissertation. University of Amsterdam.

[22] Moksh Jain and S Sowmya Kamath. 2020. Improving Convergence in IRGAN with PPO. In *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD*. 328–329.

[23] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proc. WSDM*. 781–789.

[24] Jayoung Kim, Chaejeong Lee, Yehjin Shin, Sewon Park, Minjung Kim, Noseong Park, and Jihoon Cho. 2022. Sos: Score-based oversampling for tabular data. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*. 762–772.

[25] Ouail Kitouni, Niklas Nolte, James Hensman, and Bhaskar Mitra. 2023. Disk: A diffusion model for structured knowledge. *arXiv preprint arXiv:2312.05253* (2023).

[26] Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. 2023. Tabddpm: Modelling tabular data with diffusion models. In *International conference on machine learning*. PMLR, 17564–17579.

[27] Chaejeong Lee, Jayoung Kim, and Noseong Park. 2023. Codi: Co-evolving contrastive diffusion models for mixed-type tabular synthesis. In *International Conference on Machine Learning*. PMLR, 18940–18956.

[28] Jinzhong Li, Huan Zeng, Lei Peng, Jingwen Zhu, and Zhihong Liu. 2022. Learning to rank method combining multi-head self-attention with conditional generative adversarial nets. *Array* 15 (2022), 100205.

[29] Jinzhong Li, Huan Zeng, Cunwei Xiao, Chunjuan Ouyang, and Hua Liu. 2024. Listwise learning to rank method combining approximate NDCG ranking indicator with Conditional Generative Adversarial Networks. *Pattern Recognition Letters* 179 (2024), 31–37.

[30] Jun Li, Chenyang Zhang, Wei Zhu, and Yawei Ren. 2025. A comprehensive survey of image generation models based on deep learning. *Annals of Data Science* 12, 1 (2025), 141–170.

[31] Ping Li, Qiang Wu, and Christopher Burges. 2007. Mcrank: Learning to rank using multiple classification and gradient boosting. *neurips* 20 (2007).

[32] Jianghao Lin, Yang Cao, Yong Yu, and Weinan Zhang. 2025. Diffusion Models for Recommender Systems: From Content Distribution To Content Creation. In *kdd*. 6074–6085.

[33] Jianghao Lin, Jiaqi Liu, Jiachen Zhu, Yunjia Xi, Chengkai Liu, Yangtian Zhang, Yong Yu, and Weinan Zhang. 2024. A Survey on Diffusion Models for Recommender Systems. *arXiv preprint arXiv:2409.05033* (2024).

[34] Tennison Liu, Zhaozhi Qian, Jeroen Berrevoets, and Mihaela van der Schaar. 2023. Goggle: Generative modelling for tabular data by learning relational structure. In *Proc. ICLR*.

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

[36] Shuqi Lu, Zhicheng Dou, Xu Jun, Jian-Yun Nie, and Ji-Rong Wen. 2019. Psgan: A minimax game for personalized search with limited and noisy click data. In *sigir*. 555–564.

[37] Claudio Lucchese, Franco Maria Nardini, Salvatore Orlando, R. Perego, Fabrizio Silvestri, and Salvatore Trani. 2016. Post-Learning Optimization of Tree Ensembles for Efficient Ranking. *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval* (2016). doi:10.1145/2911451.2914763

[38] Matt McGee. 2012. Yes, Bing Has Human Search Quality Raters and Here's How They Judge Web Pages. *Search Engine Land* (2012). http://searchengineland.com/bing-search-quality-rating-guidelines-130592

[39] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196* (2024).

[40] Bhaskar Mitra, Henriette Cramer, and Olya Gurevich. 2024. Sociotechnical Implications of Generative Artificial Intelligence for Information Access. In *Information Access in the Era of Generative AI*. Springer, 161–200.

[41] Bhaskar Mitra and Nick Craswell. 2018. An introduction to neural information retrieval. *Foundations and Trends® in Information Retrieval* (2018).

[42] Dae Hoon Park and Yi Chang. 2019. Adversarial sampling and training for semi-supervised information retrieval. In *www*. 1443–1453.

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

[44] Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, et al. 2024. Large language models are effective text rankers with pairwise ranking prompting. In *naacl*. 1504–1518.

[45] Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. 2008. Learning diverse rankings with multi-armed bandits. In *Proc. ICML*. 784–791.

[46] Hossein A Rahmani, Nick Craswell, Emine Yilmaz, Bhaskar Mitra, and Daniel Campos. 2024. Synthetic test collections for retrieval evaluation. In *sigir*. 2647–2651.

[47] Hossein A Rahmani, Xi Wang, Emine Yilmaz, Nick Craswell, Bhaskar Mitra, and Paul Thomas. 2025. Syndl: A large-scale synthetic test collection for passage retrieval. In *www*. 781–784.

[48] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[49] Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. 2025. TabDiff: a Mixed-type Diffusion Model for Tabular Data Generation. In *Proc. ICLR*.

[50] Ashudeep Singh and Thorsten Joachims. 2019. Policy Learning for Fairness in Ranking. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 5427–5437.

[51] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*. pmlr, 2256–2265.

[52] Yang Song and Stefano Ermon. 2019. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems* 32 (2019).

[53] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*. https://openreview.net/forum?id=PxTIG12RRHS

[54] Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agents. In *emnlp*. 14918–14937.

[55] Paul Thomas, Seth Spielman, Nick Craswell, and Bhaskar Mitra. 2024. Large language models can accurately predict searcher preferences. In *Proc. SIGIR*.

[56] Alex X Wang, Stefanka S Chukova, Colin R Simpson, and Binh P Nguyen. 2024. Challenges and opportunities of generative models on tabular data. *Applied Soft Computing* 166 (2024), 112223.

[57] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A minimax game for unifying generative and discriminative information retrieval models. In *sigir*. 515–524.

[58] Pinhuan Wang, Zhiqiu Xia, Chunhua Liao, Feiyi Wang, and Hang Liu. 2025. REALM: Recursive Relevance Modeling for LLM-based Document Re-Ranking. In *emnlp*. 23875–23889.

[59] Ryen W White and Chirag Shah. 2025. Information Access in the Era of Generative AI.

[60] Qiang Wu, Christopher JC Burges, Krysta M Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *Information Retrieval* 13, 3 (2010), 254–270.

[61] Zhen Xing, Qijun Feng, Haoran Chen, Qi Dai, Han Hu, Hang Xu, Zuxuan Wu, and Yu-Gang Jiang. 2024. A survey on video diffusion models. *Comput. Surveys* 57, 2 (2024), 1–42.

[62] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. 2019. Modeling tabular data using conditional gan. *Advances in neural information processing systems* 32 (2019).

[63] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2023. Diffusion models: A comprehensive survey of methods and applications. *Comput. Surveys* 56, 4 (2023), 1–39.

[64] Hai-Tao Yu, Degen Huang, Fuji Ren, and Lishuang Li. 2021. Diagnostic evaluation of policy-gradient-based ranking. *Electronics* 11, 1 (2021), 37.

[65] Hai-Tao Yu, Rajesh Piryani, Adam Jatowt, Ryo Inagaki, Hideo Joho, and Kyoung-Sook Kim. 2023. An in-depth study on adversarial learning-to-rank. *Information Retrieval Journal* 26, 1 (2023), 1.

[66] Yisong Yue, Thomas Finley, Filip Radlinski, and Thorsten Joachims. 2007. A support vector method for optimizing average precision. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. 271–278.

[67] Hengrui Zhang, Jiani Zhang, Balasubramaniam Srinivasan, Zhengyuan Shen, Xiao Qin, Christos Faloutso, Huzefa Rangwala, and George Karypis. 2024. Mixed-type tabular data synthesis with score-based diffusion in latent space. In *Proc. ICLR*.

[68] Shuhan Zheng and Nontawat Charoenphakdee. 2022. Diffusion models for missing value imputation in tabular data. *arXiv preprint arXiv:2210.17128* (2022).

[69] Honglei Zhuang, Zhen Qin, Kai Hui, Junru Wu, Le Yan, Xuanhui Wang, and Michael Bendersky. 2024. Beyond yes and no: Improving zero-shot llm rankers via scoring fine-grained relevance labels. In *naacl*. 358–370.

[70] Honglei Zhuang, Xuanhui Wang, Michael Bendersky, Alexander Grushetsky, Yonghui Wu, Petr Mitrichev, Ethan Sterling, Nathan Bell, Walker Ravina, and Hai Qian. 2021. Interpretable ranking with generalized additive models. In *wsdm*. 499–507.