Scaled Supervision is an Implicit Lipschitz Regularizer

Zhongyu Ouyang[1], **Chunhui Zhang**[1], **Yaning Jia**[1], **Soroush Vosoughi**[1*]

[1]Dartmouth College, Hanover, NH 03755
{zhongyu.ouyang.gr, chunhui.zhang.gr, yaning.jia.gr, soroush.vosoughi}@dartmouth.edu

Abstract

In modern social media, recommender systems (RecSys) rely on the click-through rate (CTR) as the standard metric to evaluate user engagement. CTR prediction is traditionally framed as a binary classification task to predict whether a user will interact with a given item. However, this approach overlooks the complexity of real-world social modeling, where user, item, and their interactive features change dynamically in fast-paced online environments. This dynamic nature often leads to model instability, reflected in overfitting short-term fluctuations rather than higher-level interactive patterns. While overfitting calls for more scaled and refined supervisions, current solutions often rely on binary labels that overly simplify fine-grained user preferences through the thresholding process, which significantly reduces the richness of the supervision. Therefore, we aim to alleviate the overfitting problem by increasing the supervision bandwidth in CTR training. Specifically, *(i)* theoretically, we formulate the impact of fine-grained preferences on model stability as a Lipschitz constrain; *(ii)* empirically, we discover that scaling the supervision bandwidth can act as an *implicit* Lipschitz regularizer, stably optimizing existing CTR models to achieve better generalizability. Extensive experiments show that this scaled supervision significantly and consistently improves the optimization process and the performance of existing CTR models, even without the need for additional hyperparameter tuning[1].

Introduction

In modern social media, recommender systems are ubiquitous in online applications and have redefined the user experience in product recommendation on e-commerce platforms (Wang et al. 2021; Schafer, Konstan, and Riedl 1999), personalized modelling (Gomez-Uribe and Hunt 2015; Van den Oord, Dieleman, and Schrauwen 2013; Wu et al. 2022), and friend recommendation on social media platforms (Ma et al. 2008; Jamali and Ester 2010; Fan et al. 2019; Wen et al. 2022; Ouyang et al. 2024a,b). As an important type of RecSys, click-through rate (CTR) models predict users' interaction probability, which help developers improve user engagement and recommendation accuracy in social media. CTR models specifically incorporate contextual features (e.g., user's demographic information, item descriptions, or transaction timestamps between a user-item interaction) when making interaction prediction. The problem is generally formulated as a binary classification task.

While the above CTR model training pipeline is commonly adopted, it often overlooks the underlying dynamics in the embeddings learned from the recommendation environment. Interaction prediction is delicately influenced by various factors, including recent user behavior, trends, and contextual information, all of which can shift rapidly over time. This dynamic nature poses a significant challenge for CTR models - they may struggle to keep pace with the continual changes as the binary classification formulation oversimplifies the structure of preferences. As a result, the models could overfit short-term fluctuations in the data, failing to capture general patterns that are representative of long-term interactions and impairing its stability to input noises.

The Lipschitz constant is commonly adopted to assess the stability of a model. A smaller Lipschitz constant indicates that the model's output is less prone to change drastically in response to minor variations in the input, indicating better stability. Model stability is essential for practical CTR prediction, given the fast-changing nature of inputs in RecSys. Since there are massive studies (Shi et al. 2022; Wang et al. 2024) that acknowledge accurately estimating the Lipschitz constant for a deep model is challenging due to the complexity and diversity of the feature construction layers involved, directly estimating a CTR model's Lipschitz constant is impractical. Alternatively, we study the *theoretical impact* of the granularity of preference supervisions on model stability. Our theorem suggests that training with scaled supervision via more fine-grained preferences can be formulated as an implicit Lipschitz regularizer for CTR models to achieve better stability and generalizability. In the context of CTR prediction, we propose replacing binary labels with fine-grained preference feedback to guide the training of a CTR model. With more scaled supervision, the model's output becomes smoother and more consistent, resulting in less overfitting short-term fluctuations hidden in the data with improved model generalization and stability in learning high-level interaction patterns.

In this work, we leverage the continuous nature of ratings in user feedback to increase the bandwidth of supervision signals beyond current Boolean labels in CTR prediction

*Corresponding author

[1]https://github.com/zyouyang/ImpLipReg_CTR.git

with explicit feedback. By modeling ratings categorically, we enable the learning of high-level interaction patterns across distinct rating levels in a more segmented and independent manner. These categorical ratings are subsequently transformed into binary labels, aligning with the original CTR prediction task while preserving the enriched supervision benefits. Our contributions can be summarized as follows:

- We theoretically identify the impact of the granularity of preference supervisions on a deep model's stability: more refined supervisions help reduce a model's Lipschitz constant, resulting in more stable and smoother model output.
- Inspired by the theorem, we identify the limitations of binary labeling in capturing the full spectrum of user preferences in modern social media recommender systems, and then propose to use more fine-grained preferences to increase the bandwidth of supervisions for more enhanced model stability and generalizability.
- Extensive experiments based on our open-release implementation and datasets are conducted to demonstrate that our approach significantly improves CTR prediction performance and promotes better learning dynamics in optimizing model parameters.

Preliminary

Lipschitz Constant in Deep Models

A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is said to be Lipschitz continuous on an input set $\mathcal{X} \subseteq \mathbb{R}^n$ if there exists a bound $K \geq 0$ such that for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, f satisfies:

$$\|f(\mathbf{x}) - f(\mathbf{y})\| \leq K \|\mathbf{x} - \mathbf{y}\|, \quad \forall \mathbf{x}, \mathbf{y} \in \mathcal{X}. \tag{1}$$

The smallest possible K in Equation (1) is defined as the Lipschitz constant of f, denoted as $\mathrm{Lip}(f)$:

$$\mathrm{Lip}(f) = \sup_{\mathbf{x},\mathbf{y} \in \mathcal{X}, \mathbf{x} \neq \mathbf{y}} \frac{\|f(\mathbf{x}) - f(\mathbf{y})\|}{\|\mathbf{x} - \mathbf{y}\|}. \tag{2}$$

In this context, f is known as a K-Lipschitz function, where the Lipschitz constant quantifies the maximum change in the function's output resulting from a unit-norm perturbation in its input. This constant is a crucial indicator of a deep model's stability wrt the inputs. However, determining the exact Lipschitz constant is computationally challenging. As highlighted in (Virmaux and Scaman 2018), computing the exact Lipschitz constant for deep models has been proven to be NP-hard.

CTR Prediction in Modern RecSys

General CTR Prediction Paradigm Click-through rate (CTR) prediction is a task focused on estimating the probability that a user will engage with a specific item. This prediction takes the user and item IDs, features related to the user, the item, and the broader context in which the interaction occurs as the input, and the expected output is a probability likelihood ranging from zero to one. Formally, we denote the IDs of user i and item j as x_i and x_j, respectively. We denote the raw contextual features (including user features, item features, and features related to the interaction context such as duration time) of the interaction in between as $\mathbf{c}_{ij} \in \mathbb{R}^{d^c}$, where d^c refers to the dimension of the contextual feature. The user ID x_i, item ID x_j, and the raw contextual features $\mathbf{c}_{ij}$ construct the input of the CTR prediction. The input is then encoded through the embedding layers to be transformed into dense vectors in a high-dimensional space.

Formally, we encode user/item IDs with the function $f(\cdot) : \mathbb{R} \to \mathbb{R}^d$, where d refers to the latent dimension of the encoded ID embeddings. Similarly, we encode the contextual features with the function $h(\cdot) : \mathbb{R}^{d^c} \to \mathbb{R}^{d'}$, where d' is the dimension of the encoded contextual embeddings. The encoded input $\mathbf{z}_{ij} \in \mathbb{R}^{2d+d'}$ is defined as the concatenation of the three encoded features. The above embedding process can be summarized as:

$$\mathbf{z}_i = f(x_i),\ \mathbf{z}_j = f(x_j),\ \mathbf{z}^c_{ij} = h(\mathbf{c}_{ij}), \tag{3}$$

$$\mathbf{z}_{ij} = \left[\mathbf{z}_i \parallel \mathbf{z}_j \parallel \mathbf{z}^c_{ij}\right], \tag{4}$$

where $\parallel$ refers to the concatenation operation, $\mathbf{z}_i$ and $\mathbf{z}_j$ refer to the encoded ID embeddings of user i and item j respectively, and $\mathbf{z}^c_{ij}$ are the encoded contextual features. After the embedding process, we feed the embeddings to the learnable operations to learn both low-level and high-level interaction patterns. Specifically, we denote the learnable operations as $q(\cdot) : \mathbb{R}^{2d+d'} \to \mathbb{R}$, and let $p_{ij} = q(\mathbf{z}_{ij})$, where $p_{ij} \in [0, 1]$ represents how likely user i would interact with item j. We depict the CTR prediction paradigm in Figure 1 (a).

DCN as an Examplification The learnable operations in different CTR models vary by their intrinsic model structures. To illustrate the learning scheme of CTR prediction, we here demonstrate the modeling process of DCN (Wang et al. 2021), one of the most typical CTR models commonly used in both industrial and academic environments. There are two distinct types of trainable layers in DCN: the cross layers and the deep layers, which constitute the cross and the deep network respectively. The cross-network is designed to model explicit feature interactions. Each cross-layer is formulated as

$$\mathbf{z}^{(l+1)} = \mathbf{z}^{(0)}\mathbf{z}^{(l)\intercal}\mathbf{w}^{(l)} + \mathbf{b}^{(l)} + \mathbf{z}^{(l)}, \tag{5}$$

where $\mathbf{z}^{(l+1)}, \mathbf{z}^{(l)} \in \mathbb{R}^d$ are the input and output column vectors from the l-th cross layer, and $\mathbf{w}^{(l)}, \mathbf{b}^{(l)} \in \mathbb{R}^d$ are the trainable parameters in the l-th cross layer. The deep network is a series of fully connected layers designed to capture complex and nonlinear interactions:

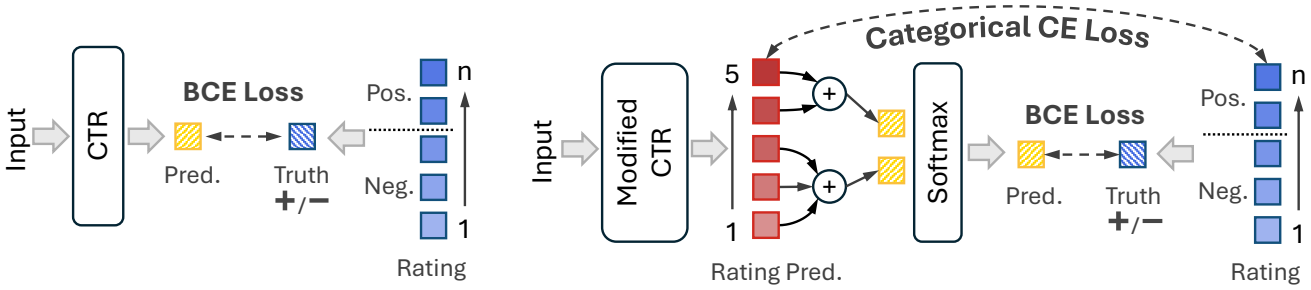
$$\mathbf{h}^{(l+1)} = \mathrm{ReLU}(\mathbf{w}^{(l+1)}\mathbf{h}^{(l)} + \mathbf{b}^{(l)}), \tag{6}$$

where $\mathbf{h}^{(l)} \in \mathbb{R}^{d_l}, \mathbf{h}^{(l+1)} \in \mathbb{R}^{d_{l+1}}$ are the input and output of the l-th hidden layer, respectively, and $\mathbf{w}^{(l)} \in \mathbb{R}^{d_{l+1} \times d_l}$ are the trainable parameters in the l-th layer.

For a given user i and item j, we let the input of the first layers as the encoded features. That is, $\mathbf{z}^{(0)} = \mathbf{h}^{(0)} = \mathbf{z}_{ij}$. The output of both networks are then concatenated and fed into a prediction layer to generate the final binary prediction:

$$p_{ij} = \sigma\left([\mathbf{z}^{(L_1)} \parallel \mathbf{h}^{(L_2)}]\mathbf{w}_{\text{logits}}\right), \tag{7}$$

where $\mathbf{z}^{(L_1)} \in \mathbb{R}^{d_1}, \mathbf{h}^{(L_2)} \in \mathbb{R}^{d_2}$ are the L_1-th and L_2-th layer outputs from the cross and deep networks, respectively,



(a) The paradigm of original CTR prediction with explicit feedback

(b) Our paradigm for CTR prediction with explicit feedback using enlarged supervision bandwidth

Figure 1: (a) and (b) demonstrate the existing and our modified paradigm of training CTR models with explicit feedback respectively. The input includes user, item IDs and the contextual information of the interaction in between. Our paradigm enlarges the supervision bandwidth from explicit preferences: we slightly modify existing CTR models' prediction layers to recover the explicit preferences for more refined supervision. These logits are then normalized via the Softmax function to the probability that matches the CTR prediction task.

$\mathbf{w}_{\text{logits}} \in \mathbb{R}^{d_1+d_2}$ is the weight vector in the logits layer, and $\sigma(x) = 1/(1+e^{-x})$. For the binary classification task formalized for the CTR prediction task, the DCN model is trained with the binary cross entropy (BCE) loss:

$$\mathcal{L} = -\frac{1}{N}\sum_{\{i,j\}\in T_r} y_{ij}\log(p_{ij}) + (1-y_{ij})\log(1-p_{ij}) + \lambda\sum_{l}\|\mathbf{w}^{(l)}\|^2, \tag{8}$$

where T_r denotes the training set of positive and negative pairs, p_{ij} is the predicted interaction probability between user i and item j, y_{ij} refers to the binary label for the user-item engagement (1 for positive and 0 for negative pairs), λ is the L_2 regularization coefficient, and $\mathbf{w}^{(l)}$ represents the parameters of the CTR model.

Estimating Lipschitz Effects of Preference Feedback

We first conduct a theoretical analysis of how the granularity of preference feedback influences the Lipschitz constant of deep models. Then, in light of our theoretical analysis, instead of seeking for a proxy to approximate the Lipschitz constant, we propose a strategy that utilizes fine-grained ratings to enhance existing CTR methods' stability and generality. Our strategy acts as an *implicit* Lipschitz regularizer and can be naturally derived without complex math derivations. It simply encourages the models to generate smoother output to mitigate the overfitting problem by capturing more general user-item interactive patterns.

More Fine-grained Supervision Enhances Model Stability

In this section, we present a theoretical foundation to explore the relationship between supervision bandwidth and model stability. Specifically, we refer the dimension of the output logits N as the supervision bandwidth and analyze how N influences the Lipschitz continuity of the model's output.

Theorem 1. *Let $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), \ldots, f_N(\mathbf{x})]$ be the logits output by a model for an input $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^D$, where N is the dimension of the output logits. Let $\mathbf{p}(\mathbf{x}) = \sigma(\mathbf{f}(\mathbf{x}))$ denote the corresponding normalized probabilities with temperature scaling. Assume that the model's logits $\mathbf{f}(\mathbf{x})$ are Lipschitz continuous with respect to $\mathbf{x}$ with Lipschitz constant L_f. Then the Lipschitz constant $L_p(N)$ of the normalization function wrt the input $\mathbf{x}$ satisfies:*

$$L_p(N) \leq \frac{L_f}{\sqrt{N}}. \tag{9}$$

Insight. Theorem 1 indicates the following key insight: as the supervision bandwidth N increases, the Lipschitz constant of the model under this finer-grained supervision decreases. In other words, as the preference feedback become more fine-grained, the norm of the Jacobian matrix is reduced, resulting in a smoother gradient landscape. Correspondingly, this reduction in the Lipschitz constant, as reflected in model gradients, decreases the sensitivity of output probabilities to input variations. This smoothing effect effectively acts as an implicit form of gradient regularization, thereby enhancing model stability (Neyshabur et al. 2019). Importantly, the benefits of this regularization grow as the complexity of the classification task increases with larger supervision bandwidth (i.e., as N becomes larger). We leave the proof in Appendix B.

Stabilizing CTR Models via Fine-grained Preference Supervision

Following our prior study, to overcome the limitations of current CTR models which rely on binary labels to provide training supervision, we propose a strategy that exploits the fine-grained nature in user feedback ratings to provide more detailed supervision signals. In the real world, explicit preference feedback is expressed through ratings $\mathbf{r}_{ij}$, where $r_{ij} \in \{1, 2, 3, ..., N\}$, reflecting varying degrees of interest

and satisfaction with item j by user i. Traditional CTR models convert these ratings into binary labels $y_{ij} \in \{0, 1\}$ based on thresholding. Interactions with ratings higher than the threshold are labeled as positive ones, and those lower than the threshold are labeled as negative ones. In our approach, we additionally adopt the fine-grained ratings as the ground truth for supervision. As shown in Figure 1 (b), our approach modifies the traditional CTR model to be supervised by the discrete ratings directly.

To adapt existing CTR models to handle these fine-grained ratings, we re-formulate the problem as a multi-class classification problem and modify their prediction layers to output explicit rating predictions. Let the modified rating prediction function be $q'(\cdot) : \mathbb{R}^{2d+d'} \rightarrow \mathbb{R}$, $\hat{r}_{ij} = q'(\mathbf{z}_{ij})$, where $\hat{r}_{ij}$ represents the predicted rating for user i and item j. The predicted ratings are then supervised with the categorical Cross Entropy (CE) loss, defined as:

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N} \sum_{\{i,j\} \in T_r} \sum_{k=1}^{K} y_{ij}^k \log p_{ij}^k, \tag{10}$$

where T_r denotes the training set with pairs of user i and item j and their labels, y_{ij}^k is the true rating label in one-hot encoding, and p_{ij}^k is the predicted probability for rating k. Note that adopting CE loss plays a crucial role in our method by heavily penalizing over-confident incorrect predictions through large logarithmic losses. This mechanism discourages skewed or overly confident logits, promoting an even distribution of logits. This aligns with the assumption of evenly distributed logits, which underpins the proof of Theorem 1. To ensure compatibility with traditional CTR tasks, the explicit rating predictions are then aggregated into a probability that matches the binary prediction. Let $\hat{p}_{ij}$ be the aggregated probability that user i will click on item j, defined as

$$\hat{p}_{ij} = \frac{\sum_{k > t_{sh}} p_{ij}^k}{\sum_{k \leq t_{sh}} p_{ij}^k + \sum_{k > t_{sh}} p_{ij}^k}, \tag{11}$$

where p_{ij}^k is the predicted probability for rating k and t_{sh} is the rating threshold defining the positive and negative feedback. This aggregated probability $\hat{p}_{ij}$ can then be used in the traditional BCE loss for final supervision:

$$\begin{aligned} \mathcal{L}_{\text{BCE}} = &- \frac{1}{N} \sum_{\{i,j\} \in T_r} y_{ij} \log(\hat{p}_{ij}) + (1 - y_{ij}) \log(1 - \hat{p}_{ij}) \\ &+ \lambda \sum_{l} \|\hat{\mathbf{w}}^{(l)}\|^2, \end{aligned} \tag{12}$$

where y_{ij} is the binary label indicating whether user i clicked on item j, $\hat{\mathbf{w}}^{(l)}$ represents the parameters of the modified CTR model. The final supervised loss $\mathcal{L}$ is defined as:

$$\mathcal{L} = \lambda_r \mathcal{L}_{CE} + (1 - \lambda_r) \mathcal{L}_{BCE}, \tag{13}$$

where $\lambda_r \in [0, 1]$ is the contributing ratio of the categorical cross-entropy loss.

Boosting user engagement from the perspective of behavioral psychology

Prior research in behavioral psychology examines the relationship between user engagement and recommender systems (Pachali and Datta 2024; Zhang, Wang, and Ariffin 2024), showing that platform-driven features (e.g., playlists on Spotify) significantly boost interaction. Additionally, factors like user satisfaction and dependence play crucial roles in sustaining engagement. Our approach, which stabilizes CTR models for improved generality, aligns with these findings enabling the models to better capture and respond to user preferences. This is accomplished by focusing on learning general user behaviors rather than overfitting to noisy interactions, such as bait clicks or non-engaged interactions. From the end user's perspective, our method results in more personalized and accurate recommendations, reducing exposure to irrelevant content and providing a more satisfying and tailored experience. Enhanced user satisfaction and engagement not only improve the user experience but also deliver substantial commercial and psychological benefits, creating a mutually beneficial outcome for both users and companies. We show a concrete example for end user experience improvement in recommendation in Appendix A.4.

Experiments

We evaluate the predictive and ranking performance of applying our strategy to multiple popular baselines used in social media RecSys. To assess stability, we compare the gradient norms before and after applying our approach to a representative CTR model. Additionally, we demonstrate that our method consistently outperforms popular baselines, even without exhaustive hyperparameter tuning. Lastly, we report the average computational time before and after integrating our approach into the baselines to demonstrate its superior practicability to real-world applications.

Setup

Datasets We select three publicly available and popular recommendation benchmark datasets to conduct the experiments: *(1)* **MovieLens-1M**[2] dataset is sourced from the MovieLens website, containing 1 million ratings from 6,000 users on 4,000 movies; *(2)* **Yelp2018**[3] dataset includes a large collection of user reviews, item information, and user-item interactions from the Yelp platform, with millions of ratings, reviews, and other data points such as item categories and user profiles; *(3)* **Amazon-Book**[4] dataset is derived from the Amazon product data and contains detailed information on user interactions with books, including millions of ratings, reviews, and metadata such as book categories, titles, and author information. Based on data sparsity, we categorize ML-1M as a dense dataset, and Yelp2018 and Amazon-Book are identified as sparse datasets. The dataset statistics are provided in Appendix B. The feedback provided by users in the above datasets is explicit, with the format of ratings from

[2]https://grouplens.org/datasets/movielens/1m/

[3]https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset/

[4]http://jmcauley.ucsd.edu/data/amazon/

Dataset	*ML-1M*		*Yelp2018*		*Amazon-book*	
Metric	Orig.	Smooth (Δ%)	Orig.	Smooth (Δ%)	Orig.	Smooth (Δ%)
Model	**AUC (%) ↑**					
WideDeep	82.10	82.46 (0.44)	74.59	74.96 (0.49)	80.95	81.42 (0.58)
NFM	81.82	82.02 (0.25)	74.12	74.55 (0.57)	80.73	81.22 (0.60)
xDeepFM	81.38	82.43 (1.29)	74.55	74.89 (0.46)	80.95	81.39 (0.54)
AutoInt	82.09	82.56 (0.58)	74.57	74.99 (0.56)	80.97	81.37 (0.50)
FiGNN	81.58	81.66 (0.10)	74.59	74.89 (0.40)	80.96	81.25 (0.36)
DCNV2	82.16	82.61 (0.55)	74.43	74.78 (0.47)	80.89	81.27 (0.47)
EulerNet	82.03	82.00 (−0.04)	74.54	74.78 (0.32)	80.97	81.26 (0.36)
Model	**LogLoss (×100) ↓**					
WideDeep	50.83	50.50 (−0.65)	55.20	55.02 (−0.32)	39.01	38.57 (−1.11)
NFM	51.28	51.30 (0.04)	55.56	55.23 (−0.60)	39.23	38.92 (−0.79)
xDeepFM	52.07	51.14 (−1.79)	55.28	55.03 (−0.46)	39.04	39.76 (1.85)
AutoInt	51.05	50.36 (−1.34)	55.27	55.04 (−0.42)	38.97	38.70 (−0.70)
FiGNN	52.04	51.90 (−0.27)	55.28	55.36 (0.14)	39.07	39.03 (−0.08)
DCNV2	51.04	50.43 (−1.19)	55.32	55.18 (−0.25)	39.01	38.87 (−0.36)
EulerNet	51.39	51.62 (0.45)	55.21	55.04 (−0.31)	38.97	39.28 (0.79)

Table 1: Performance of CTR models on three datasets before and after applying our smoothing strategy. **Orig.** denotes baseline results, and **Smooth** (Δ%) shows results after smoothing with relative improvement or degradation in parentheses.

1 to 5. For all the feedback, we convert the ratings to binary labels through thresholding and set the threshold as 3. We randomly split datasets with a ratio of 0.8/0.1/0.1 for training, validation, and testing, respectively.

Baselines We select seven CTR baselines that specifically focus on feature interaction modeling: WideDeep (Cheng et al. 2016), NFM (He and Chua 2017), xDeepFM (Lian et al. 2018), AutoInt (Song et al. 2019), FiGNN (Li et al. 2019), DCNV2 (Wang et al. 2021), and EulerNet (Tian et al. 2023b). Focusing on models designed specifically for feature interaction modeling isolates the impact of supervision bandwidth enlargement effectively. Including diverse model types with differing objectives (e.g., behavior prediction, auxiliary tasks) could introduce modeling biases, confounding the true effects of our supervision bandwidth modification. Specifically: NFM (He and Chua 2017) and xDeepFM (Lian et al. 2018) combine the advantages of Factorization Machines (Rendle 2010) and deep neural networks (DNNs) to capture complex non-linear and high-order feature interactions. DCNV2 (Wang et al. 2021) learns explicit and implicit feature interactions through a cross-network and a DNN, respectively. It improves DCN with a low-rank cross-network that enhances the efficiency and interpretability of the model. AutoInt (Song et al. 2019) utilizes self-attentive neural networks to learn more effective feature interactions. EulerNet (Tian et al. 2023b) learns high-order feature interactions by transforming their exponential powers into linear combinations of the modulus and phase of complex features.

Training configurations For all the baselines, we employ the AdamW (Loshchilov and Hutter 2019) optimizer for optimization. We run a fixed number of grid searches over all the baseline models' provided hyper-parameters for their best AUC performance on the validation set. With consistent hyper-parameters, we train the models under five random seeds and save all the checkpoints. We repeat the hyper-parameter search process for our method as well. The models' CTR prediction abilities are mainly evaluated by two predictive metrics, AUC (↑) and logloss (↓). In addition, we evaluate the ranking abilities of the CTR models, i.e., how well the models rank the more relative/highly rated items. We do so to emphasize that the recommendation task is essentially a ranking task, therefore it is crucial to preserve the ranking abilities of the CTR models. For the ranking metrics, we select NDCG@K and recall@k, and set k to 10 and 20. We adopt the recommender system library named RecBole (Zhao et al. 2021) to conduct all the experiments.

Improved predictive and ranking performance

Predictive Improvement For all the baseline models, we first evaluate their recommendation performance on the three benchmark datasets by the predictive metrics, AUC (the higher the better) and logloss (the lower the better). The results are shown in Table 1, where the **Orig.** column shows the performance of the original CTR models, the **Smooth** column shows the performance of applying our method to the corresponding CTR models, and Δ% represents the relative improvement wrt the original performance. From the table, we observe that: *(i)* While the degree of improvement varies across the combination of CTR models and benchmark datasets, our strategy stably improves the AUC over the original model. On average, we observe ∼0.47% AUC improvement. The improvement is particularly significant for the CTR prediction task compared to prior methods focused on enhancing feature interaction modeling. For instance, in terms of AUC on the Movielens dataset, DCNV2 (Wang et al. 2021), AFN (Cheng, Shen, and Huang 2020), and FinalNet (Zhu et al. 2023) achieve relative gains of approximately

Model / Metric	NDCG@10 ↑ Orig.	NDCG@10 ↑ Smooth (Δ%)	NDCG@20 ↑ Orig.	NDCG@20 ↑ Smooth (Δ%)	Recall@10 ↑ Orig.	Recall@10 ↑ Smooth (Δ%)	Recall@20 ↑ Orig.	Recall@20 ↑ Smooth (Δ%)
(i) ML-1M								
WideDeep	11.70	11.72 (0.21)	14.12	14.34 (1.56)	13.49	13.86 (2.73)	21.17	21.92 (3.51)
NFM	11.90	12.34 (3.72)	14.57	15.11 (3.66)	14.21	15.04 (5.85)	22.60	23.55 (4.21)
xDeepFM	11.97	13.65 (14.09)	14.62	16.24 (11.09)	14.31	15.67 (9.52)	22.63	24.11 (6.53)
AutoInt	11.16	12.24 (9.73)	13.68	14.78 (8.10)	13.06	14.08 (7.78)	20.91	22.09 (5.65)
FiGNN	11.48	12.19 (6.17)	14.21	14.86 (4.53)	14.10	14.66 (3.96)	22.45	22.93 (2.12)
DCNV2	11.54	13.16 (14.07)	14.01	15.66 (11.79)	13.52	14.93 (10.45)	21.21	22.97 (8.32)
EulerNet	11.58	12.58 (8.69)	14.33	15.17 (5.82)	14.11	14.79 (4.88)	22.46	22.96 (2.20)
(ii) Yelp2018								
WideDeep	3.05	3.14 (3.22)	5.29	5.38 (1.66)	6.15	6.34 (3.19)	14.52	14.66 (0.92)
NFM	3.78	3.97 (4.92)	6.16	6.38 (3.70)	7.57	7.89 (4.23)	16.29	16.78 (2.98)
xDeepFM	3.36	3.05 (−9.11)	5.65	5.27 (−6.80)	6.63	6.21 (−6.28)	15.11	14.45 (−4.35)
AutoInt	2.99	3.22 (7.56)	5.28	5.52 (4.62)	6.15	6.55 (6.40)	14.67	15.12 (3.03)
FiGNN	3.09	3.16 (2.26)	5.37	5.49 (2.16)	6.28	6.48 (3.09)	14.76	15.12 (2.45)
DCNV2	2.99	2.95 (−1.20)	5.35	5.25 (−1.87)	6.29	6.21 (−1.34)	15.07	14.75 (−2.11)
EulerNet	4.44	4.33 (−2.57)	6.73	6.67 (−0.98)	8.11	8.01 (−1.21)	16.55	16.64 (0.51)
(iii) Amazon-book								
WideDeep	2.42	2.71 (12.24)	3.76	4.16 (10.53)	4.08	4.69 (15.11)	8.69	9.57 (10.06)
NFM	2.49	2.47 (−0.88)	3.82	3.89 (1.88)	4.19	4.37 (4.25)	8.73	9.20 (5.38)
xDeepFM	3.36	3.05 (−9.11)	3.60	3.60 (0.00)	3.84	4.05 (5.25)	8.31	8.74 (5.20)
AutoInt	2.38	2.68 (12.77)	3.76	4.13 (9.73)	4.07	4.63 (13.75)	8.83	9.51 (7.70)
FiGNN	2.38	2.54 (6.72)	3.72	3.96 (6.45)	3.97	4.43 (11.54)	8.58	9.23 (7.65)
DCNV2	2.83	2.89 (2.19)	4.53	4.64 (2.61)	5.10	5.40 (5.88)	11.02	11.44 (3.79)
EulerNet	2.38	3.05 (28.24)	3.76	4.46 (18.66)	4.07	4.97 (21.94)	8.83	9.71 (10.04)

Table 2: The ranking results of the CTR models before and after applying our implicit Lipschitz regularizer. **Orig.** represents the base model, and **Smooth** (Δ%) denotes performance after smoothing with relative improvement in parentheses.

0.1%, 0.05%, and 0.07% over the previous state-of-the-art, respectively. The improved AUC results validate that our strategy is effective in improving the CTR model's ability to accurately identify items of interest, across various model structures and benchmark datasets. *(ii)* The values of logloss are generally decreased, suggesting more stable and confident predictions. In some rare cases, we observe slight increments in the logloss. However, this slight increase is acceptable with the corresponding improved AUC, which is directly associated with the quality of the recommendation.

Ranking Improvement In addition to the predictive metrics, we also assess the impact of our strategy on the CTR models' ranking capabilities, which is ranking more relevant items higher than the less relevant ones. Although predictive metrics such as AUC and logloss are well-aligned with the CTR prediction task, they do not fully capture the practical nuances in real-world recommendations. One thing to constantly bear in mind is that recommendation is fundamentally a ranking task. In practice, the predictive probabilities of a CTR model are commonly used as the ranking score to rank recommended items. Thus, while enhancing the predictive ability of a CTR model, it is equally important to ensure that its ranking performance is well-preserved. We compare the ranking metrics of CTR models before and after applying our approach and demonstrate the results in Table 2.

From Table 2, we observe that our strategy not only maintains the ranking capabilities of the original CTR models, but also improves them. While there are a few instances of performance degradation, the overall improvements are remarkable. Our strategy acts as a regularizer, resulting in a more balanced distribution of probabilities across rating classes. This smoother prediction reduces the risk of overconfidence by preventing excessively high ranking scores for certain user-item pairs, thus improving the model's ability to distinguish between relevant and irrelevant items and leading to more refined ranking lists. Although the ranking metrics are not directly optimized by our auxiliary objective (i.e., categorical loss), the Lipschitz regularization statistically improves ranking performance. This improvement is significant even for the Yelp2018 dataset, which includes a wide variety of businesses such as restaurants, hotels, shopping malls, and services. The broader scope of Yelp2018 introduces more variability and complexity compared to the other two datasets.

Enlarging preference supervision bandwidth as an *implicit Lipschitz regularizer*

Embeddings in recommender systems are designed to represent users and items as low-dimensional vectors that encapsulate their unique characteristics. Given the dynamic

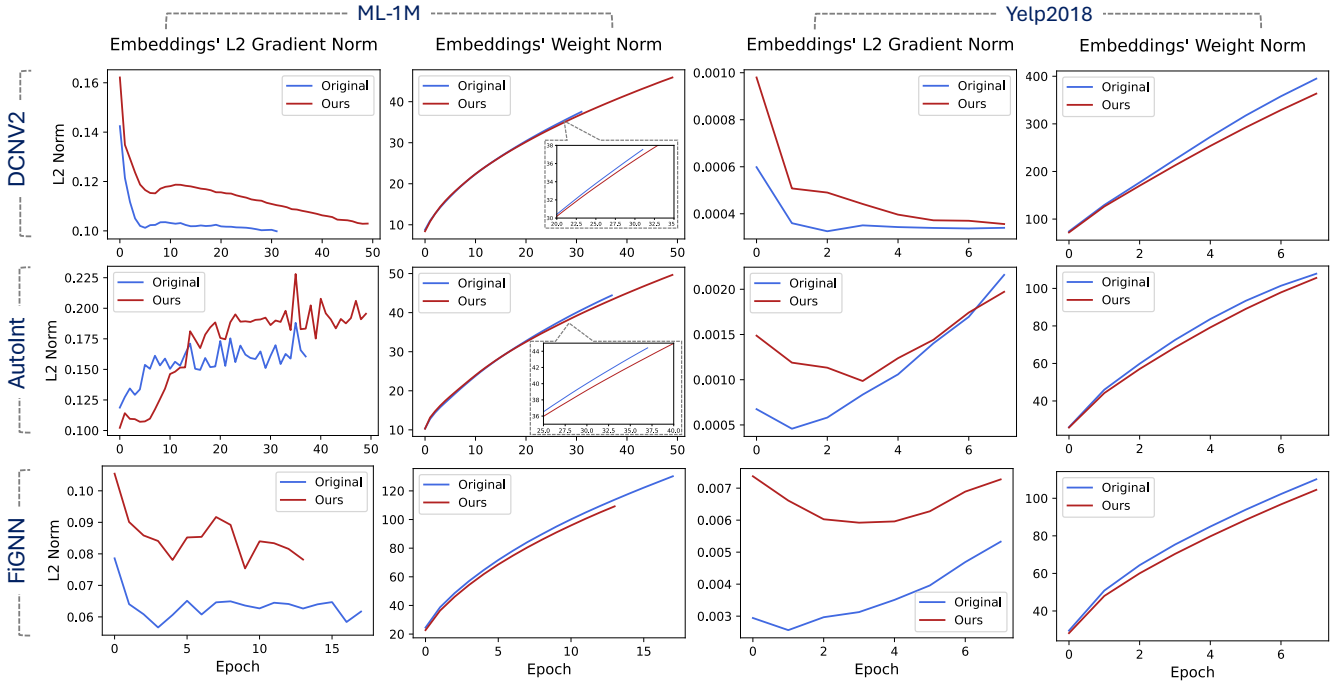


Figure 2: The L2 gradient norms during training and the weight norms post-training for the learned user and item embeddings in the **original** DCNV2, AutoInt, and FiGNN methods, and those after applied with **our** approach.

nature of recommendation, where user preferences and item characteristics change frequently, these embeddings should be promptly updated to capture real-time changes in user-item interaction patterns. Especially, frequent embedding updates help the model better adapt to rapidly evolving data, ensuring that personalized recommendations remain relevant. However, frequent embedding updates present a challenge in optimizing both the embeddings and the subsequent learnable operations, such as MLPs and cross-layers. While these operations are designed to capture high-level interactions between users and items, they can be prone to overfitting. The fluctuations in embedding updates, driven by ever-changing user-item interactions, can cause the learnable operations to become overly sensitive to short-term data patterns, reducing their ability to generalize across the global feature space.

To address this issue, it is crucial to encourage more active and distributed adjustments in representation learning to keep up with the dynamics in recommendation. To verify that our strategy effectively foster active representation learning, we analyze the dynamics of embedding updates during training and their properties post-training. Specifically, we examine the gradient norms of embeddings to assess the extent of updates made during training, providing insights into how responsive the embeddings are to input changes. The larger the gradient norms are, the more active the embeddings are updated. Furthermore, we analyze the weight norms of the embeddings to demonstrate the compactness and generality, which are properties critical to mitigate overfitting and ensuring model stability. Embeddings with smaller weight norms are considered more compact and generalized. In this experiment, we select DCNV2 (Wang et al. 2021), AutoInt (Song et al. 2019), and FiGNN (Li et al. 2019) as the representative CTR models and analyze their respective L2 gradient norms during training and weight norms post-training in ML-1M (relatively dense) and Yelp2018 (relatively sparse). The results are shown in Figure 2.

From the figure, we observe: *(i)* Compared to the respective CTR models, applying our strategy consistently results in higher gradient norms, indicating more active embedding updates during training; *(ii)* The weight norms of embeddings are uniformly reduced with our approach, reflecting enhanced embedding compactness. Embeddings with lower weight norms exhibit less sensitivity to input noise, thereby improving model robustness and generalizability; *(iii)* in contrast to the sparse dataset Yelp2018, in the dense dataset ML-1M, our method achieves a greater reduction in the embeddings' L2 gradient norms while exhibiting a smaller reduction in weight norms. This is because dense datasets like ML-1M provide rich interaction signals, enabling our approach to leverage abundant information for more active embedding updates, resulting in numerically larger gradient norms. Conversely, the noisy interaction signals prevalent in sparse datasets like Yelp2018 lead our method to prioritize stabilizing embeddings through reduced weight norms.

Implicit Lipschitz regularizer improves model robustness towards noisy perturbations

Real-world datasets often contains varying levels of input noises. For example, a user may adjust their rating of an item based on mainstream media reviews, or intentionally modifies

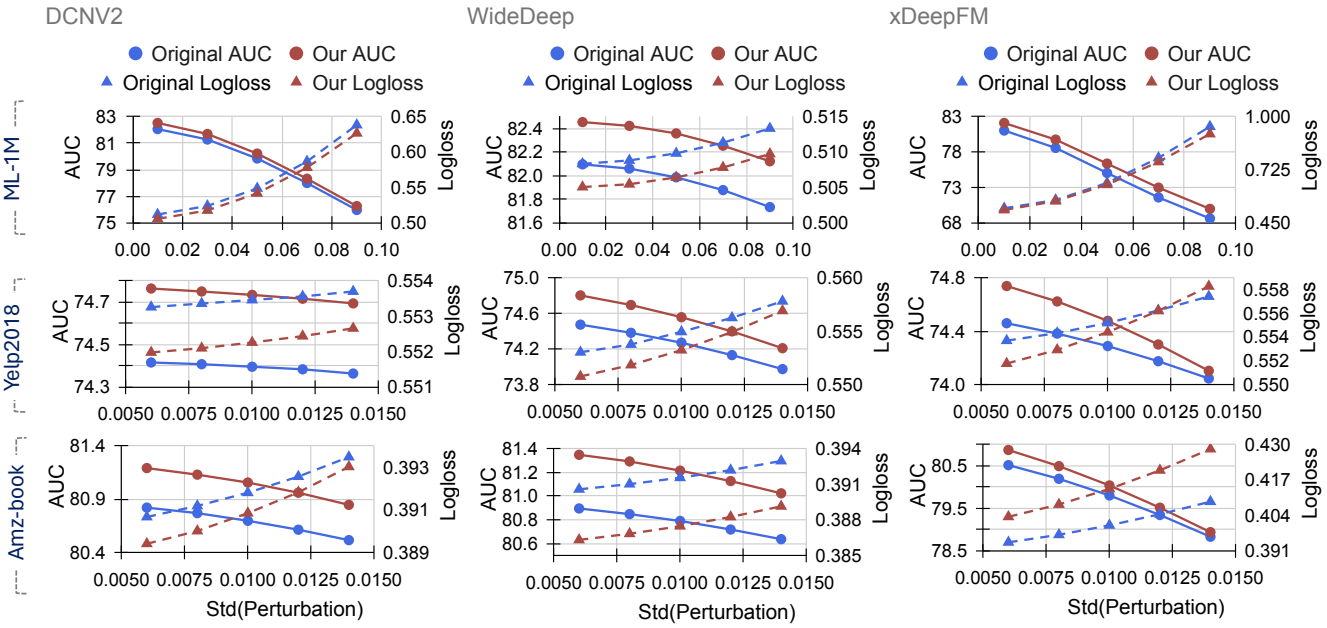


Figure 3: The changes in AUC and logloss when perturbations are applied to the input embeddings of the **original** CTR methods and those after applying **our** method. Left y-axis represents AUC (↑) and right y-axis corresponds to logloss (↓).

their profile information to prevent privacy leakage. A robust CTR model should be able to handle such noise without compromising its ability to capture general user preferences. To verify that our approach improves model robustness against such noises, we introduce random Gaussian perturbations to the embeddings, with the standard deviation of the perturbations controlling the noise magnitude. We conduct experiments using DCNV2 (Wang et al. 2021), WideDeep (Cheng et al. 2016), and xDeepFM (Lian et al. 2018) as the representative CTR models. For the dense dataset ML-1M, we vary standard deviation across $\{0.01, 0.03, 0.05, 0.07, 0.09\}$; for the sparse datasets Yelp2018 and Amazon-book, the standard deviations are chosen from $\{0.006, 0.008, 0.01, 0.012, 0.014\}$ due to the higher sensitivity of sparse datasets to noise. The smaller noise magnitudes for sparse datasets are necessary, as the sparsity of interactions amplifies the impact of noise to model's generality. We show the results in Figure 3.

The figure illustrates that our method exhibits superior robustness by consistently achieving higher AUCs and lower logloss values across all the selected datasets and models. In dense datasets like ML-1M, the inherent data density reduces its sensitivity to perturbations, enabling our method to consistently outperform the original methods even when the performance has decreased by approximately 9%. Conversely, in sparse datasets, our method gradually loses its advantage, as sparse interactions amplify sensitivity to noisy embeddings. This is because more generalized embeddings may struggle to recover the original information under larger perturbations. Nevertheless, our approach's ability to outperform the original methods under realistic noise levels in sparse datasets underscores its overall superior robustness and reliability.

Implicit Lipschitz regularizer achieves coefficient-invariant improvement

Our strategy only introduces the coefficient of the categorical cross-entropy loss $\lambda_r \in [0, 1]$ as the hyperparameter. To verify the model's performance sensitivity to λ_r, we select DCNV2 (Wang et al. 2021), WideDeep (Cheng et al. 2016), and xDeepFM (Lian et al. 2018) as the representative models, apply our strategy to the models with varied value for λ_r in $\{0.1, 0.3, 0.5, 0.7, 0.9\}$, and compare the AUC performance with those of the original ones. For each setting, we repeat the experiment under five seeds and report the average performance over the five seeds. The comparison is shown in Figure 4. From the figure, we see that our strategy consistently improves the AUC score of the original model: with all the λ_r values selected, our strategy yields at least 0.07% and at most 0.65% performance improvement. This phenomenon suggests that our way of increasing the supervision bandwidth not only aligns well with the original CTR task, but also regularizes the learning to achieve better model generalization ability. Moreover, while no consistent pattern emerges across models and datasets for selecting the optimal value of λ_r, the results in the figure suggest that the maximum performance gain is typically achieved when λ_r is approximately 0.7. Hence, in practice, initializing λ_r at 0.7 in our method serves as a reasonable starting point, likely to deliver near-optimal performance.

Implicit Lipschitz regularizer enhances model generalizability with minimally increased latency

Since our implicit regularizer configures the CTR model to produce fine-grained preference predictions in order to take

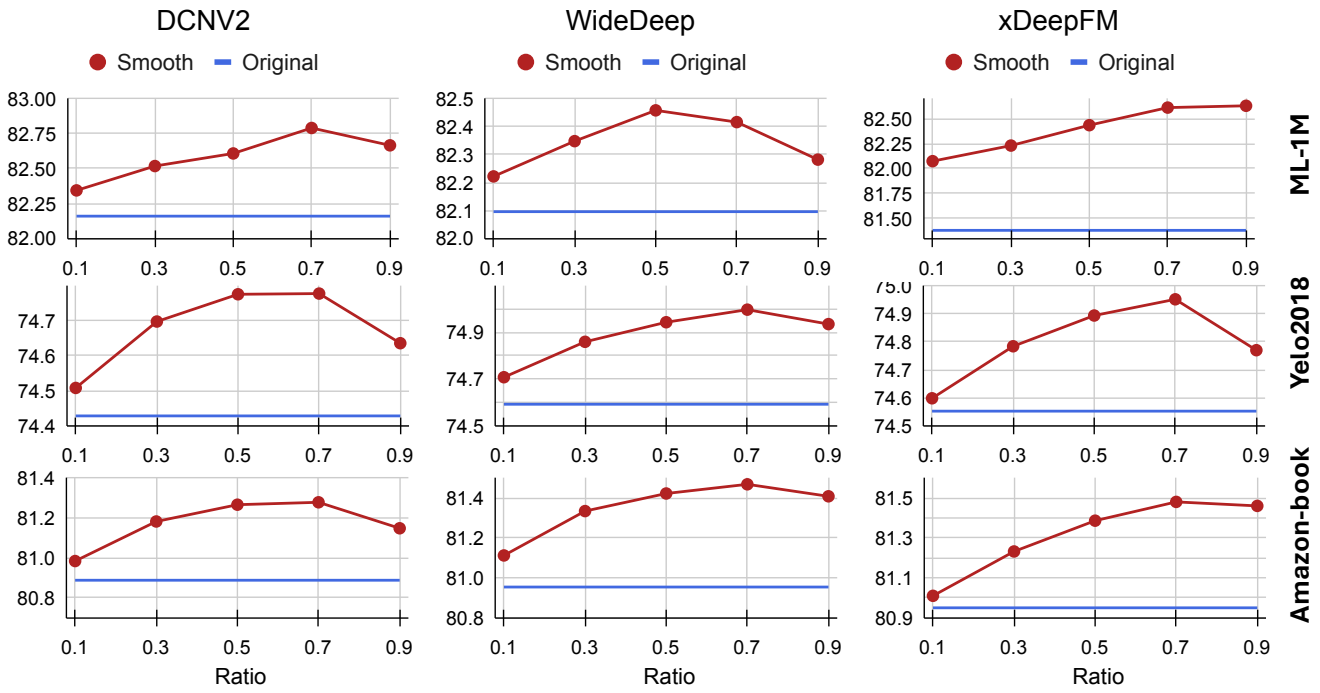


Figure 4: The AUC score of DCNV2 (Wang et al. 2021), WideDeep (Cheng et al. 2016), and xDeepFM (Lian et al. 2018) on the ML-1M, Yelp2018, and Amazon-book dataset with different ratio λ_r in training.

advantage of the increased supervision bandwidth, we additionally examine whether/how much additional time cost is associated with the performance improvement. We verify the extent of the time increment of our strategy by logging the time for *each training epoch*, *the complete training process*, and *the evaluation process*. For each benchmark dataset, we average the corresponding time for the selected CTR models in seconds and show the results in Table 3. From the table, we see that the relative time increment is not prominent. While it takes slightly longer to train a modified CTR model relative to the original one, the increased relative time for evaluation is rather minimal (i.e., less than $\sim 0.02\%$). This advantage suggests our strategy's superior practicability to existing industrial recommendation pipelines, as no significant latency is introduced for prominent performance improvement.

Phase	Dataset	Orig.	Smooth (Δ%)
Training Epoch	*ML-1M*	1.78	2.64 (48.31)
	Yelp2018	5.42	6.00 (10.70)
	Amz.-book	12.44	12.47 (0.24)
Full Training	*ML-1M*	73.68	77.47 (5.14)
	Yelp2018	43.37	51.02 (17.66)
	Amz.-book	108.62	147.79 (36.01)
Eval.	*ML-1M*	0.12	0.13 (8.33)
	Yelp2018	0.29	0.30 (3.45)
	Amz.-book	0.48	0.50 (4.17)

Table 3: The time (in seconds) for each training epoch, the full training process, and evaluation on NVIDIA RTX 3090 GPU. **Orig.** is the original CTR model, **Smooth** (Δ%) is the result after applying our method with the relative change in percentage shown in parenthesis.

Related Work

Click-through Rate Prediction. CTR prediction aims to estimate the likelihood of a user interacting with an item, given features such as user/item IDs and contextual signals (e.g., demographics, timestamps). Existing research primarily focuses on modeling feature interactions and user behavior, with auxiliary methods explored for further gains. Early models like Factorization Machines (FM) (Rendle 2010) laid the groundwork for modeling low-order feature interactions. Deep learning extensions such as NFM (He and Chua 2017), DeepFM (Guo et al. 2017), and xDeepFM (Lian et al. 2018) enable both low- and high-order interactions via hybrid architectures. Other works such as DCN (Wang et al. 2017), AFN (Cheng, Shen, and Huang 2020), AutoInt (Song et al. 2019), and DCNV2 (Wang et al. 2021), introduce specialized modules like cross networks to further enhance feature modeling (Vaswani et al. 2017; Mei et al. 2024).

On the other hand, behavior modeling approaches focus on user interest dynamics. DIN (Zhou et al. 2018) and DIEN (Zhou et al. 2019) apply attention over historical interactions, while sequential models like DPN (Zhang et al. 2024) and pretrained methods like SRP4CTR (Han et al. 2024) emphasize long-term preference learning. Furthermore, Auxiliary strategies such as contrastive learning

(e.g., CL4CTR (Wang et al. 2023)) and pretraining (e.g., SRP4CTR (Han et al. 2024)) have shown promise for improved representations. Recent architectural improvements such as FinalNet (Zhu et al. 2023) and FinalMLP (Mao et al. 2023), aim to optimize efficiency and feature expressiveness.

Unlike prior work, we enhance CTR performance by modifying the supervision signal rather than model architecture. Building on existing feature-interaction models, we revise only the prediction heads to align with fine-grained preference signals. This lightweight change maintains model structure while encouraging smoother outputs and better generalization.

Lipschitz Constant in Deep Models. The Lipschitz constant is crucial for characterizing the stability of neural networks, especially in convolutional and attention-based models (Zou, Balan, and Singh 2019; Kim, Papamakarios, and Mnih 2021; Araujo et al. 2021). Several works propose Lipschitz normalization for GNNs (Dasoulas, Scaman, and Virmaux 2021; Jia, Zhang, and Vosoughi 2024; Yuan et al. 2024a), while others (Das et al. 2023) explore relaxing uniform Lipschitz constraints for privacy and optimization. Further, Gama and Sojoudi (2022) estimate filter Lipschitz bounds to stabilize graph models, and Ye et al. (2023) propose a Lipschitz-regularized transformer (LRFormer) to counter overconfidence in vision transformers. However, most traditional methods require costly matrix verification procedures (Xu and Sivaranjani 2024).

In contrast, our approach implicitly enforces Lipschitz regularization by utilizing fine-grained preference feedback. Rather than explicitly computing Lipschitz constants, our method adaptively promotes smooth predictions that fit the underlying data distribution, leading to improved robustness and generalization in CTR models. We provide a more detailed related work in Appendix B.

Conclusion

In this paper, we focus on the model overfitting problem in the context of CTR prediction. We first point out that most CTR models, especially their embeddings, cannot keep pace with the fast-changing recommendation environment where user preferences and item characteristics change frequently. As a result, they inevitably overfit the short-term fluctuations in the data rather than the high-level interactions. Then theoretically, we discover that increasing the supervision bandwidth reduces the model's Lipschitz constant, which inversely measures a model's stability with the input. Empirically, following the above theorem, we propose to enlarge the supervision bandwidth by recovering the refined rating preferences during the model training process in addition to the existing binary supervisions. Through extensive experiments, we verify that our strategy: (i) generally improves both the predictive and rating performance of CTR models; (ii) decouples the learning of embeddings and the learnable operations (e.g., MLPs, cross layers, etc.) to alleviate overfitting; (iii) achieves consistent performance improvement with any reasonable coefficient; (iv) exhibits superior practicability with minimal extra evaluation latency.

Limitations

While our study provides both theoretical and empirical evidence that increasing supervision bandwidth enhances model stability and generalizability, we recognize several limitations: *(i)* Each CTR model possesses its own distinct architecture. Determining the most effective way to modify these structures to support the expanded supervision scheme is an open problem, as there may be multiple approaches to adjust each CTR model. Developing comprehensive guidelines tailored to these architectural adaptations is crucial for fully harnessing the benefits of augmented supervision and achieving optimal model performance. Nevertheless, our current implementation of modifying only prediction layers only is simple yet effective. *(ii)* Expanding the supervision bandwidth necessitates access to more granular preference feedback. However, identifying fine-grained ground truth becomes challenging in contexts where such detailed feedback is not readily available. Interaction metrics such as clicks, dwell time, and other forms of engagement do not inherently convey the nuances of user preferences. Therefore, further research is needed to explore methods for applying our strategy to CTR models trained primarily on implicit feedback, identifying alternative ways to infer and incorporate fine-grained preferences. We extend the discussion in Appendix A.1.

Acknowledgments

This research was supported in part by the National Science Foundation under Grant No. 2452367.

References

Araujo, A.; Negrevergne, B.; Chevaleyre, Y.; and Atif, J. 2021. On Lipschitz regularization of convolutional layers using toeplitz matrix theory. In *AAAI Conference on Artificial Intelligence*.

Cheng, H.-T.; Koc, L.; Harmsen, J.; Shaked, T.; Chandra, T.; Aradhye, H.; Anderson, G.; Corrado, G.; Chai, W.; Ispir, M.; et al. 2016. Wide & deep learning for recommender systems. In *1st Workshop on deep learning for recommender systems*.

Cheng, W.; Shen, Y.; and Huang, L. 2020. Adaptive factorization network: Learning adaptive-order feature interactions. In *The AAAI Conference on Artificial Intelligence*.

Das, R.; Kale, S.; Xu, Z.; Zhang, T.; and Sanghavi, S. 2023. Beyond uniform lipschitz condition in differentially private optimization. In *International Conference on Machine Learning*.

Dasoulas, G.; Scaman, K.; and Virmaux, A. 2021. Lipschitz normalization for self-attention layers with application to graph neural networks. In *International Conference on Machine Learning*.

Fan, W.; Ma, Y.; Li, Q.; He, Y.; Zhao, E.; Tang, J.; and Yin, D. 2019. Graph neural networks for social recommendation. In *ACM Web Conference*.

Gama, F.; and Sojoudi, S. 2022. Distributed linear-quadratic control with graph neural networks. *Signal Processing*.

Gomez-Uribe, C. A.; and Hunt, N. 2015. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems*.

Gu, A.; and Dao, T. 2024. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. In *First Conference on Language Modeling*.

Guo, H.; Tang, R.; Ye, Y.; Li, Z.; and He, X. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *International Joint Conference on Artificial Intelligence*.

Guo, Z.; Zhang, C.; Fan, Y.; Tian, Y.; et al. 2023. Boosting Graph Neural Networks via Adaptive Knowledge Distillation. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence*.

Han, R.; Li, Q.; Jiang, H.; Li, R.; Zhao, Y.; Li, X.; and Lin, W. 2024. Enhancing CTR Prediction through Sequential Recommendation Pre-training: Introducing the SRP4CTR Framework. In *ACM International Conference on Information and Knowledge Management*.

He, X.; and Chua, T.-S. 2017. Neural factorization machines for sparse predictive analytics. In *ACM SIGIR Conference*.

Jamali, M.; and Ester, M. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *ACM Recommender Systems conference*.

Jia, Y.; Zhang, C.; and Vosoughi, S. 2024. Aligning Relational Learning with Lipschitz Fairness. In *Proceedings of the 2024 International Conference on Learning Representations*.

Kim, H.; Papamakarios, G.; and Mnih, A. 2021. The Lipschitz constant of self-attention. In *International Conference on Machine Learning*.

Li, Z.; Cui, Z.; Wu, S.; Zhang, X.; and Wang, L. 2019. Fi-gnn: Modeling feature interactions via graph neural networks for ctr prediction. In *ACM international conference on information and knowledge management*.

Lian, J.; Zhou, X.; Zhang, F.; Chen, Z.; Xie, X.; and Sun, G. 2018. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *ACM SIGKDD Conference*.

Lin, J.; Chen, B.; Wang, H.; Xi, Y.; Qu, Y.; Dai, X.; Zhang, K.; Tang, R.; Yu, Y.; and Zhang, W. 2024. ClickPrompt: CTR Models are Strong Prompt Generators for Adapting Language Models to CTR Prediction. In *The World Wide Web Conference*.

Liu, Z.; Zhang, C.; Tian, Y.; Zhang, E.; Huang, C.; Ye, Y.; et al. 2023. Fair Graph Representation Learning via Diverse Mixture-of-Experts. In *Proceedings of the 2023 International World Wide Web Conference*.

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

Ma, H.; Yang, H.; Lyu, M. R.; and King, I. 2008. Sorec: social recommendation using probabilistic matrix factorization. In *ACM International Conference on Information and Knowledge Management*.

Mao, K.; Zhu, J.; Su, L.; Cai, G.; Li, Y.; and Dong, Z. 2023. FinalMLP: an enhanced two-stream MLP model for CTR prediction. In *AAAI Conference on Artificial Intelligence*.

Mei, L.; Cheng, Y.; Liu, R.; Yin, Z.; Jiang, W.; Wang, S.; and Gong, W. 2024. ESP-PCT: Enhanced VR Semantic Performance through Efficient Compression of Temporal and Spatial Redundancies in Point Cloud Transformers. In *International Joint Conference on Artificial Intelligence*.

Miyato, T.; Kataoka, T.; Koyama, M.; and Yoshida, Y. 2018. Spectral Normalization for Generative Adversarial Networks. In *International Conference on Learning Representations*.

Müller, R.; Kornblith, S.; and Hinton, G. E. 2019. When does label smoothing help? *Advances in Neural Information Processing Systems*.

Neyshabur, B.; Li, Z.; Bhojanapalli, S.; LeCun, Y.; and Srebro, N. 2019. The role of over-parametrization in generalization of neural networks. In *International Conference on Learning Representations*.

Ouyang, Z.; Zhang, C.; Hou, S.; Ma, S.; Chen, C.; Li, T.; Xiao, X.; Zhang, C.; and Ye, Y. 2024a. Symbol Prompt Tuning Completes the App Promotion Graph. In *Proceedings of the 2024 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*.

Ouyang, Z.; Zhang, C.; Hou, S.; Zhang, C.; and Ye, Y. 2024b. How to Improve Representation Alignment and Uniformity in Graph-based Collaborative Filtering? In *Proceedings of the 2024 International AAAI Conference on Web and Social Media*.

Pachali, M. J.; and Datta, H. 2024. What drives demand for playlists on Spotify? *Marketing Science*.

Qian, Y.; Zhang, C.; Zhang, Y.; Wen, Q.; Ye, Y.; et al. 2022. Co-Modality Imbalanced Graph Contrastive Learning. In *Proceedings of the Thirty-sixth Conference on Neural Information Processing Systems*.

Qiu, Q.; Zhu, T.; Gong, H.; Chen, L.; and Ning, H. 2024. ReLU-KAN: New Kolmogorov-Arnold Networks that Only Need Matrix Addition, Dot Multiplication, and ReLU. *arXiv preprint arXiv:2406.02075*.

Rendle, S. 2010. Factorization machines. In *IEEE International Conference on Data Mining*.

Schafer, J. B.; Konstan, J.; and Riedl, J. 1999. Recommender systems in e-commerce. In *ACM conference on Electronic commerce*.

Shi, Z.; Wang, Y.; Zhang, H.; Kolter, J. Z.; and Hsieh, C.-J. 2022. Efficiently Computing Local Lipschitz Constants of Neural Networks via Bound Propagation. In *Advances in Neural Information Processing Systems*.

Song, W.; Shi, C.; Xiao, Z.; Duan, Z.; Xu, Y.; Zhang, M.; and Tang, J. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *ACM International Conference on Information and Knowledge Management*.

Sun, Y.; Li, X.; Dalal, K.; Xu, J.; Vikram, A.; Zhang, G.; Dubois, Y.; Chen, X.; Wang, X.; Koyejo, S.; et al. 2024. Learning to (learn at test time): Rnns with expressive hidden states. *arXiv preprint arXiv:2407.04620*.

Tian, Y.; Dong, K.; Zhang, C.; and Zhang, C. 2023a. Heterogeneous Graph Masked Autoencoders. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence*.

Tian, Z.; Bai, T.; Zhao, W. X.; Wen, J.-R.; and Cao, Z. 2023b. EulerNet: Adaptive Feature Interaction Learning via Euler's Formula for CTR Prediction. In *ACM SIGIR Conference*.

Van den Oord, A.; Dieleman, S.; and Schrauwen, B. 2013. Deep content-based music recommendation. In *Advances in Neural Information Processing Systems*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*.

Virmaux, A.; and Scaman, K. 2018. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *Advances in Neural Information Processing Systems*.

Wang, F.; Wang, Y.; Li, D.; Gu, H.; Lu, T.; Zhang, P.; and Gu, N. 2023. Cl4ctr: A contrastive learning framework for ctr prediction. In *ACM Conference on Web Search and Data Mining*.

Wang, H.; Zhao, M.; Xie, X.; Li, W.; and Guo, M. 2019. Knowledge graph convolutional networks for recommender systems. In *The World Wide Web Conference*.

Wang, R.; Fu, B.; Fu, G.; and Wang, M. 2017. Deep and cross network for ad click predictions. In *Workshop on Data Mining for Online Advertising*.

Wang, R.; Shivanna, R.; Cheng, D.; Jain, S.; Lin, D.; Hong, L.; and Chi, E. 2021. Dcn v2: Improved deep and cross network and practical lessons for web-scale learning to rank systems. In *ACM Web Conference*.

Wang, Z.; Hu, B.; Havens, A. J.; Araujo, A.; Zheng, Y.; Chen, Y.; and Jha, S. 2024. On the Scalability and Memory Efficiency of Semidefinite Programs for Lipschitz Constant Estimation of Neural Networks. In *International Conference on Learning Representations*.

Wen, Q.; Ouyang, Z.; Zhang, C.; Qian, Y.; Zhang, C.; and Ye, Y. 2024. GCVR: Reconstruction from Cross-View Enable Sufficient and Robust Graph Contrastive Learning. In *Proceedings of the 40th Conference on Uncertainty in Artificial Intelligence*.

Wen, Q.; Ouyang, Z.; Zhang, J.; Qian, Y.; Ye, Y.; and Zhang, C. 2022. Disentangled dynamic heterogeneous graph learning for opioid overdose prediction. In *Proceedings of the 2022 ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Wu, J.; Zhang, C.; Liu, Z.; Zhang, E.; Wilson, S.; et al. 2022. GraphBERT: Bridging Graph and Text for Malicious Behavior Detection on Social Media. In *Proceedings of the 2022 IEEE International Conference on Data Mining*.

Xu, Y.; and Sivaranjani, S. 2024. ECLipsE: Efficient Compositional Lipschitz Constant Estimation for Deep Neural Networks. In *Advances in Neural Information Processing Systems*.

Ye, W.; Ma, Y.; Cao, X.; and Tang, K. 2023. Mitigating transformer overconfidence via Lipschitz regularization. In *Uncertainty in Artificial Intelligence*.

Yuan, X.; Tian, Y.; Zhang, C.; Ye, Y.; Chawla, N. V.; et al. 2024a. Graph Mixed Supervised Learning via Generalized Knowledge. In *Proceedings of the 2024 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Yuan, X.; Zhang, C.; Tian, Y.; Ye, Y.; and Zhang, C. 2024b. Mitigating Emergent Robustness Degradation on Graphs while Scaling Up. In *Proceedings of the 2024 International Conference on Learning Representations*.

Yue, H.; Zhang, C.; Zhang, C.; and Liu, H. 2022. Label-invariant Augmentation for Semi-Supervised Graph Classification. In *Proceedings of the Thirty-sixth Conference on Neural Information Processing Systems*.

Zhang, C.; Huang, C.; Li, Y.; Zhang, X.; Ye, Y.; and Zhang, C. 2022. Look Twice as Much as You Say: Scene Graph Contrastive Learning for Self-Supervised Image Caption Generation. In *Proceedings of the 2022 ACM International Conference on Information and Knowledge Management*.

Zhang, C.; Huang, C.; Tian, Y.; Wen, Q.; Ouyang, Z.; Li, Y.; Ye, Y.; et al. 2023a. When Sparsity Meets Contrastive Models: Less Data Can Bring Better Class-Balanced Representations. In *Proceedings of the Fortieth International Conference on Machine Learning*.

Zhang, C.; Liu, H.; Li, J.; Ye, Y.; and Zhang, C. 2023b. Mind the Gap: Mitigating the Distribution Gap in Graph Few-shot Learning. *Transactions on Machine Learning Research*.

Zhang, C.; Tian, Y.; Ju, M.; Liu, Z.; Ye, Y.; Chawla, N.; et al. 2023c. Chasing All-Round Graph Representation Robustness: Model, Training, and Optimization. In *Proceedings of the 11th International Conference on Learning Representations*.

Zhang, H.; Pan, J.; Liu, D.; Jiang, J.; and Li, X. 2024. Deep Pattern Network for Click-Through Rate Prediction. In *ACM SIGIR Conference*.

Zhang, Q.; Wang, Y.; and Ariffin, S. K. 2024. Keep scrolling: An investigation of short video users' continuous watching behavior. *Information & Management*.

Zhao, W. X.; Mu, S.; Hou, Y.; Lin, Z.; Chen, Y.; Pan, X.; Li, K.; Lu, Y.; Wang, H.; Tian, C.; et al. 2021. Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In *ACM International Conference on Information and Knowledge Management*.

Zhou, G.; Mou, N.; Fan, Y.; Pi, Q.; Bian, W.; Zhou, C.; Zhu, X.; and Gai, K. 2019. Deep interest evolution network for click-through rate prediction. In *AAAI Conference on Artificial Intelligence*.

Zhou, G.; Zhu, X.; Song, C.; Fan, Y.; Zhu, H.; Ma, X.; Yan, Y.; Jin, J.; Li, H.; and Gai, K. 2018. Deep interest network for click-through rate prediction. In *ACM SIGKDD Conference*.

Zhu, J.; Jia, Q.; Cai, G.; Dai, Q.; Li, J.; Dong, Z.; Tang, R.; and Zhang, R. 2023. Final: Factorized interaction layer for ctr prediction. In *ACM SIGIR Conference*.

Zou, D.; Balan, R.; and Singh, M. 2019. On Lipschitz bounds of general convolutional neural networks. *IEEE Transactions on Information Theory*.

Paper Checklist

1. For most authors...

(a) Would answering this research question advance science without violating social contracts, such as violating privacy norms, perpetuating unfair profiling, exacerbating the socio-economic divide, or implying disrespect to societies or cultures? Yes, and our research aligns with the core principles of ethical research and social responsibility.

(b) Do your main claims in the abstract and introduction accurately reflect the paper's contributions and scope? Yes

(c) Do you clarify how the proposed methodological approach is appropriate for the claims made? Yes

(d) Do you clarify what are possible artifacts in the data used, given population-specific distributions? No, because the data used in this work reflects personal preferences.

(e) Did you describe the limitations of your work? Yes, in the Limitations section.

(f) Did you discuss any potential negative societal impacts of your work? No, because this work mainly discuss how to improve model generalizability, which leads to more accurate user preference prediction.

(g) Did you discuss any potential misuse of your work? No, because the work is intended to improve model generalizability and stability.

(h) Did you describe steps taken to prevent or mitigate potential negative outcomes of the research, such as data and model documentation, data anonymization, responsible release, access control, and the reproducibility of findings? No, because the data does not contain personalized information other than the interaction histories. The users and items are anonymized.

(i) Have you read the ethics review guidelines and ensured that your paper conforms to them? Yes.

2. Additionally, if your study involves hypotheses testing...

(a) Did you clearly state the assumptions underlying all theoretical results? NA

(b) Have you provided justifications for all theoretical results? NA

(c) Did you discuss competing hypotheses or theories that might challenge or complement your theoretical results? NA

(d) Have you considered alternative mechanisms or explanations that might account for the same outcomes observed in your study? NA

(e) Did you address potential biases or limitations in your theoretical framework? NA

(f) Have you related your theoretical results to the existing literature in social science? NA

(g) Did you discuss the implications of your theoretical results for policy, practice, or further research in the social science domain? NA

3. Additionally, if you are including theoretical proofs...

(a) Did you state the full set of assumptions of all theoretical results? NA

(b) Did you include complete proofs of all theoretical results? NA

4. Additionally, if you ran machine learning experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? Yes, the data is public and the codes and are provided in the anonymous URL.

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? Yes, in section Experimental Settings.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? No, because the variances are minimal and we omit it following previous works.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? Yes, in section Experimental Settings.

(e) Do you justify how the proposed evaluation is sufficient and appropriate to the claims made? Yes, in section Comparison with CL-based Methods.

(f) Do you discuss what is "the cost" of misclassification and fault (in)tolerance? No, because under the case of recommendation, there is not an absolutely right or wrong decision.

5. Additionally, if you are using existing assets (e.g., code, data, models) or curating/releasing new assets, **without compromising anonymity**...

(a) If your work uses existing assets, did you cite the creators? Yes, in section Experimental Settings.

(b) Did you mention the license of the assets? Yes, the licenses are mentioned in the cited papers.

(c) Did you include any new assets in the supplemental material or as a URL? Yes, we provide it in an anonymous URL.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? No, because the data is publicly available for research purposes.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? No, because the data is anonymized.

(f) If you are curating or releasing new datasets, did you discuss how you intend to make your datasets FAIR? NA

(g) If you are curating or releasing new datasets, did you create a Datasheet for the Dataset? NA

6. Additionally, if you used crowdsourcing or conducted research with human subjects, **without compromising anonymity**...

(a) Did you include the full text of instructions given to participants and screenshots? NA

(b) Did you describe any potential participant risks, with mentions of Institutional Review Board (IRB) approvals? NA

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? NA

(d) Did you discuss how data is stored, shared, and deidentified? NA

A Discussions

A.1 Explicit Preference Feedback Availability

While our method requires explicit preference feedback to leverage its full potential, the availability of such feedback is not a significant barrier. Many widely used datasets already provide fine-grained preference feedback in the form of ratings or similar metrics. Examples include but not limit to Movielens[5], Book-Crossing[6], Anime[7]. All of the datasets are extensively used in assessing research studies in recommender systems (Wang et al. 2019; Lin et al. 2024). Even in cases where fine-grained feedback is unavailable, reasonable proxies are often available and can be employed. For instance, the Frappe[8] dataset includes information on the number of times an app has been used by a user, where a larger number of usage times indicates a stronger preference. Similarly, the Last.FM[9] dataset provides listening counts, which are indicative of user preferences. These proxy statistics, while not explicit ratings, offer sufficient granularity to be utilized as a substitute for direct feedback.

Lastly, it is crucial to reiterate the broader aim of this work: addressing the limitations of current practices in processing fine-grained feedback in state-of-the-art CTR prediction models. Many existing methods (Wang et al. 2019; Lin et al. 2024) rely on thresholding to convert fine-grained feedback into binary labels, inherently losing valuable preference information. This paper advocates for an alternative approach to better utilize fine-grained feedback, emphasizing the need for a paradigm shift in how such data is perceived and processed. Encouraging the collection and use of more fine-grained feedback could lead to significant advancements in the field.

A.2 Beyond CTR Prediction

Our method addresses the issue of overfitting in CTR prediction by leveraging fine-grained preference feedback, which is often underutilized by recent works due to limitations introduced by thresholding. This approach is specifically designed to accommodate the dynamic nature of recommendation and is tailored for binary classification tasks where such information loss prominently contributes to overfitting. For recommendation tasks that also involve binary classification and suffer from the same loss of information due to thresholding, our method is expected to be directly applicable. However, for tasks that do not meet these criteria, modifications to the method would be necessary to align with the task's unique requirements. At its core, our method's strength lies in its focus on stabilizing models through increased supervision bandwidth. This principle is not limited to CTR prediction and can be adapted to other suitable tasks, provided that the method is properly modified to address the specific challenges of those tasks. This adaptability underscores the broader applicability of the approach to enhancing model stability across various recommendation scenarios.

A.3 Applicability to Other Structures

The theoretical foundation of increasing supervision bandwidth as a regularization method suggests it is broadly applicable to various neural network architectures, including those with specialized structures or non-ReLU activation functions, such as Mamba (Gu and Dao 2024), KAN (Qiu et al. 2024), TTT (Sun et al. 2024), and boarder GNN-based methods (). The core idea hinges on stabilizing the learning process by controlling the Lipschitz constant, which is not inherently tied to a specific activation function or network structure. However, the effectiveness of this approach in such networks depends on how supervision bandwidth interacts with the specific characteristics of these architectures, such as their activation dynamics and structural constraints. For instance, non-ReLU activations might exhibit different sensitivities to overfitting or instability, requiring adjustments to the supervision strategy. In principle, the method is adaptable, but practical application to these architectures would require empirical validation and potentially task-specific modifications to optimize performance while maintaining theoretical guarantees.

A.4 From the End-user Experience

Our study demonstrates that leveraging fine-grained supervision signals enhances not only ranking metrics, such as NDCG and Recall, but also predictive metrics in CTR models. These improvements translate directly into better user experiences across various stages of interaction with recommendation systems: (i) higher ranking metrics ensure that the *most relevant items consistently appear at the top of recommendation lists*, fostering trust and reducing cognitive effort during decision-making. For example, users browsing for recommended movies are more likely to find their preferred movies prominently displayed, creating an efficient and satisfying movie recommendation experience. Improved Recall further broadens the range of relevant items presented, enabling users to discover items they might not have explicitly demonstrate preferences, such as items liked by similar users. This enhances engagement and supports serendipitous discoveries, enriching the overall user journey; (ii) In addition to ranking, the improved predictive metrics ensure that the *recommendations more accurately reflect user preferences*, leading to greater personalization. For instance, with a higher CTR prediction rate, the system can prioritize items that users are more likely to engage with. This refinement improves the ranking compared to the original model by making the top-listed items not only relevant but also highly actionable, reducing irrelevant or marginally useful recommendations. By enhancing both predictive and ranking metrics, our method supports the overarching goal of delivering recommendation systems that perform well quantitatively while providing

[5]https://grouplens.org/datasets/movielens/

[6]https://grouplens.org/datasets/book-crossing/

[7]https://www.kaggle.com/datasets/CooperUnion/anime-recommendations-database

[8]http://baltrunas.info/research-menu/frappe

[9]https://grouplens.org/datasets/hetrec-2011/

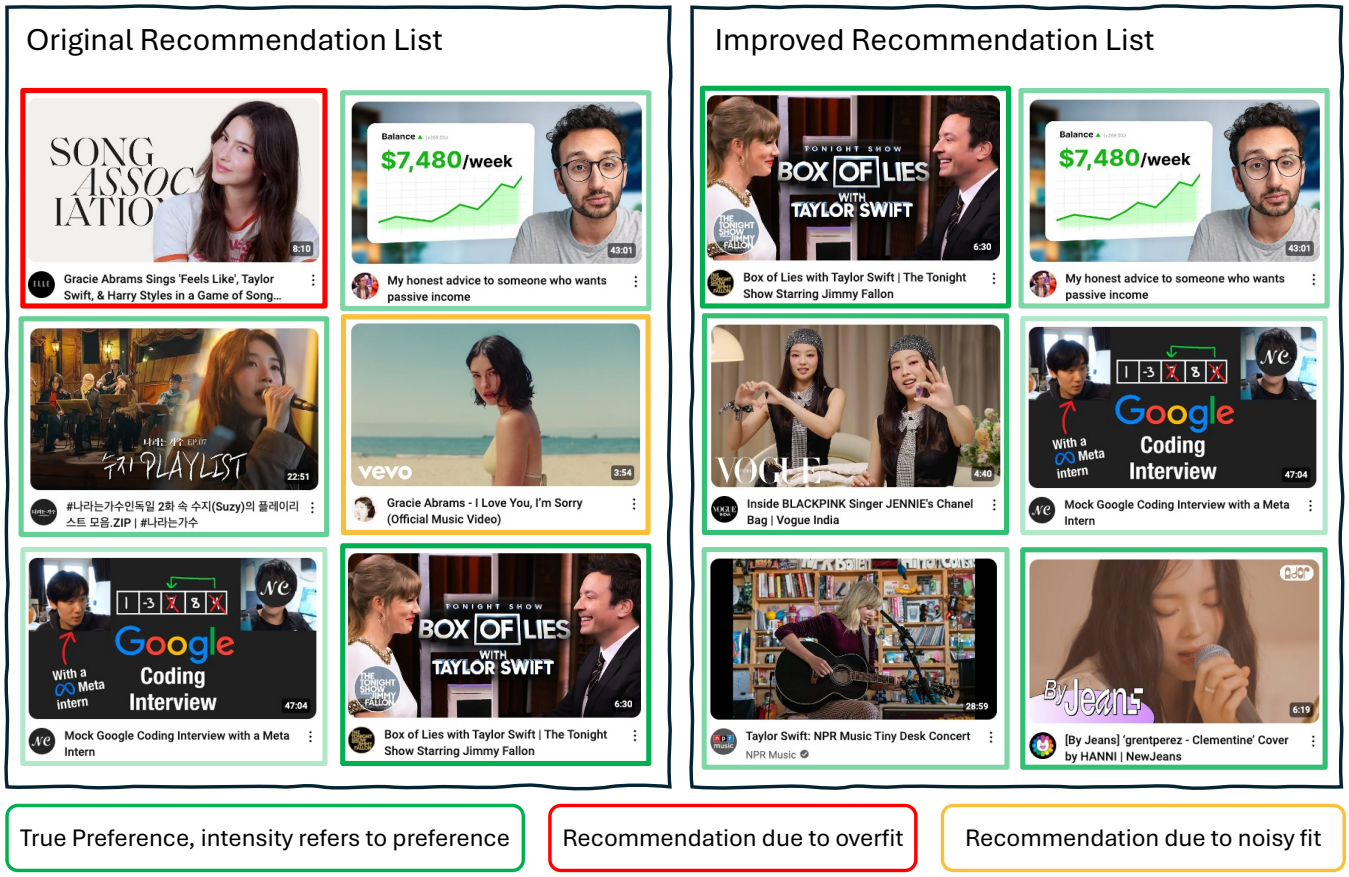


Figure 5: A concrete example for end user experience improvement in recommendation. Compared with the recommendation list generated by the original model (left side), with improved generality, our models generates recommended items (right side) that is both accurate (i.e., contents are preferred) and personalized (i.e., more interested items are ranked in the front).

meaningful, satisfying, and personalized user experiences. We show a concrete example for the enhanced user experience in Figure 5.

A.5 Stabilizing Relational Learning and their Applications: Beyond the Lipschitz Methods

Stability in relational learning models, especially in GNN and their application in recommendation systems, has been studied from multiple perspectives. A prominent direction involves explicitly regularizing the Lipschitz constant to enhance robustness and generalizability. For instance, Yuan et al. (2024b) identifies an emergent robustness degradation phenomenon in graph representation learning when models are scaled up, addressing it through a generalized knowledge distillation framework. Similarly, Wen et al. (2024) improves graph contrastive learning by reconstructing representations from cross-view information, implicitly stabilizing representations against data perturbations. Zhang et al. (2023a) further demonstrates that sparsity-driven contrastive models not only reduce data requirements but also enhance representation stability, addressing class imbalance problems. Zhang et al. (2022) generalizes the relational learning to the image captioning tasks, which uses self-supervised contrastive learning to stabilize the image captioning training. Qian et al. (2022) uses self-supervised contrastive learning to stably learn to process relational data such as AMiner. Tian et al. (2023a) explores the mask autoencoder's effect to ease the relational graph model's training.

Several other approaches have directly aimed at enhancing all-around robustness in graph learning through model architecture, training methods, and optimization techniques. Zhang et al. (2023c) presents an integrated framework for robust graph representation learning by comprehensively modeling and mitigating various robustness risks. Meanwhile, Zhang et al. (2023b) tackles the critical challenge of distribution gaps in graph few-shot learning, showing improved model stability and performance by explicitly bridging these gaps. Liu et al. (2023) extends fairness concerns into a unified framework, proposing diverse mixture-of-experts architectures to learn fair and stable graph representations, which have broader societal implications.

Alongside these explicit Lipschitz and architectural methods, recent research also leveraged knowledge distillation to stably improve the generalization of GNNs. Guo et al. (2023) proposes an adaptive knowledge distillation approach, refining the learning objectives of student models through

teacher-student interactions, thus improving GNN generalization and robustness. Similarly, Yue et al. (2022) employs label-invariant data augmentation strategies, enhancing semi-supervised graph classification performance and robustness.

B Proof of Theorem 1

Proof. Let $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), \ldots, f_N(\mathbf{x})]$ be the logits output by the model where N is the dimension of the output logits. The softmax normalization function with the built-in temperature scaling is defined as:

$$p_i(\mathbf{x}) = \frac{\exp\left(f_i(\mathbf{x})/\tau\right)}{\sum_{j=1}^{N} \exp\left(f_j(\mathbf{x})/\tau\right)} \quad \text{for } i = 1, 2, \ldots, N, \tag{14}$$

where $\tau > 0$ is the temperature parameter that adjusts the uniform distribution. We aim to derive a relation on the Lipschitz constant $L_p(N)$ of the temperature-scaled probabilities with respect to the input $\mathbf{x}$. The Lipschitz constant is defined as the smallest $L_p(N)$ such that for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$,

$$\|\mathbf{p}(\mathbf{x}) - \mathbf{p}(\mathbf{y})\| \leq L_p(N)\|\mathbf{x} - \mathbf{y}\|. \tag{15}$$

The sensitivity of the model can be analyzed by considering the Jacobian matrix of the model output probabilities with the output logits. The Jacobian $\mathbf{J}_\sigma(\mathbf{f})$ is an $N \times N$ matrix:

$$\frac{\partial p_i(\mathbf{x})}{\partial f_j(\mathbf{x})} = p_i(\mathbf{x})(\delta_{ij} - p_j(\mathbf{x})), \tag{16}$$

where δ_{ij} is the Kronecker delta. The Frobenius norm of this Jacobian matrix provides a measure of the sensitivity of the model output to changes in the logits:

$$\|\mathbf{J}_\sigma(\mathbf{f})\|_F = \sqrt{\sum_{i=1}^{N}\sum_{j=1}^{N} \left(p_i(\mathbf{x})(\delta_{ij} - p_j(\mathbf{x}))\right)^2}. \tag{17}$$

When the logits are evenly distributed as N is increasing, p_i approaches $\frac{1}{N}$ for all i, aligning closely with insights from label smoothing (Müller, Kornblith, and Hinton 2019). Then, the elements of the Jacobian matrix can be estimated as:

$$\frac{\partial p_i}{\partial f_j} \approx \frac{1}{N}(\delta_{ij} - \frac{1}{N}), \tag{18}$$

and the Frobenius norm becomes:

$$\begin{aligned}\|\mathbf{J}_\sigma(\mathbf{f})\|_F \approx &\sqrt{N \cdot \left(\frac{1}{N}\left(1 - \frac{1}{N}\right)\right)^2} \\ &+ N(N-1) \cdot \left(\frac{1}{N^2}\right)^2.\end{aligned} \tag{19}$$

The dominant term for large N is:

$$\|\mathbf{J}_\sigma(\mathbf{f})\|_F \approx \frac{1}{\sqrt{N}}. \tag{20}$$

Given that the logits $\mathbf{f}(\mathbf{x})$ are Lipschitz continuous with Lipschitz constant L_f, the sensitivity of the model output with respect to the input $\mathbf{x}$ can be constrained by:

$$\begin{aligned}\|\mathbf{p}(\mathbf{x}) - \mathbf{p}(\mathbf{y})\| &\leq \|\mathbf{J}_\sigma(\mathbf{f})\|_F \cdot \|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{y})\| \\ &\leq \|\mathbf{J}_\sigma(\mathbf{f})\|_F \cdot L_f\|\mathbf{x} - \mathbf{y}\|.\end{aligned} \tag{21}$$

The Frobenius norm of the model Jacobian matrix is constrained by:

$$\|\mathbf{J}_\sigma(\mathbf{f})\|_F \leq \frac{1}{\sqrt{N}}, \tag{22}$$

as each component $p_i(\mathbf{x})(\delta_{ij} - p_j(\mathbf{x}))$ is small when the logits are comparable, and the summation across N terms distributes the influence (Miyato et al. 2018). Thus, the Lipschitz constant $L_p(N)$ of the model function satisfies:

$$L_p(N) \leq \frac{L_f}{\sqrt{N}}. \tag{23}$$

□

Data Statistics

The statistics of the benchmark datasets are shown in Table 4.

Dataset	#User	#Item	#Inter	Sparsity%
ML-1M	6K	3K	1M	94.93
Yelp2018	77K	46K	2M	99.94
Amazon-book	68K	66K	3M	99.93

Table 4: The statistics of the benchmark datasets.

Related Work

Click-through Rate Prediction The problem of click-through rate (CTR) prediction is defined as predicting the interaction likelihood between a user and an item given the user ID, item ID, and optional context features. The incorporated contextual information includes user demographic features, item descriptions, interaction timestamps, and any other information related to the user-item interaction. These features additionally consider the variability of user preferences for items across different contexts in which they interact with the system. Recent efforts in improving CTR prediction performance can be broadly categorized into methods focusing on feature interaction modeling, user behavior modeling, with some works exploring auxiliary methods to further enhance prediction performance.

Models focus on feature interaction aim to uncover the relationships between various input features, such as user attributes, item characteristics, and contextual factors. Early works like Factorization Machines (FM) (Rendle 2010) laid the foundation for capturing low-order feature interactions efficiently by modeling pairwise relationships through factorization techniques. Building on this, recent methods incorporate deep learning to capture both low-order and high-order interactions. For example, hybrid models such as Neural Factorization Machines (NFM) (He and Chua 2017) and DeepFM (Guo et al. 2017) combine factorization techniques with DNNs to learn implicit and explicit feature interactions simultaneously. Others, like xDeepFM (Lian et al. 2018) and Deep & Cross Network (DCN) (Wang et al. 2017), introduce specialized architectures to explicitly capture feature crosses while leveraging the power of deep networks. Adaptive Factorization Network (AFN) (Cheng, Shen, and Huang 2020)

further extends this line of work by employing logarithmic neural transformations to adaptively learn high-order feature interactions in sparse feature spaces. Innovations such as AutoInt (Song et al. 2019) and DCNV2 (Wang et al. 2021) further refine this process by utilizing self-attention mechanisms and its variants (Vaswani et al. 2017; Mei et al. 2024) and improved scalability for large-scale applications.

Another significant direction focuses on understanding user interests, particularly in scenarios involving sequential or dynamic behaviors. These methods aim to adaptively model user preferences based on historical interactions and evolving contexts. Approaches like the Deep Interest Network (DIN) (Zhou et al. 2018) and its extensions, such as the Deep Interest Evolution Network (DIEN) (Zhou et al. 2019), use attention mechanisms to capture the relevance of historical user actions to the current context. Sequential models like Deep Pattern Network (DPN) (Zhang et al. 2024) and pretraining-based methods like SRP4CTR (Han et al. 2024) emphasize learning long-term user preferences and behavior patterns to enhance personalization.

Recent research has also explored auxiliary techniques, such as contrastive learning and pretraining, to improve the generalization and robustness of CTR models. For example, CL4CTR (Wang et al. 2023) introduces a contrastive learning framework to refine feature representations, while methods like SRP4CTR (Han et al. 2024) leverage sequential recommendation pretraining to boost CTR prediction performance. Structural innovations, such as those in FinalNet (Zhu et al. 2023) and FinalMLP (Mao et al. 2023), focus on improving training efficiency and feature interaction learning through optimized neural architectures.

Unlike the previous work, our work improves the performance of CTR prediction by shifting the attention to the bandwidth of supervision signals. Instead of modifying the architecture or framework of the CTR models, upon the existing CTR methods focusing on feature interaction modeling, we remain their perspective model architectures and only modify the prediction heads to match the enlarged supervision through fine-grained preference signals in the data. Such minimal structure modifications keep the model functionality intact while encouraging the model to yield smoother preference scores, which result in stronger generalizability and better performance.

Lipschitz Constant in Deep Models. The Lipschitz constant has emerged as a critical concept for understanding and controlling the stability and robustness of deep models. Existing studies have extensively explored its applications in models leveraging convolutional and attention mechanisms (Zou, Balan, and Singh 2019; Kim, Papamakarios, and Mnih 2021; Araujo et al. 2021). For example, Dasoulas, Scaman, and Virmaux (2021); Jia, Zhang, and Vosoughi (2024); Yuan et al. (2024a) propose Lipschitz normalization in graph attention networks. Das et al. (2023) explores relaxing the uniform Lipschitz condition to improve optimization efficiency and privacy-utility trade-offs in the study of differential privacy. More recently, Gama and Sojoudi (2022) estimated filter Lipschitz bounds using the infinity norm of matrices, contributing to stabilize graph-based neural models. Ye et al. (2023) introduced a Lipschitz Regularized Transformer (LR-Former) to address overconfidence in transformer models within computer vision tasks. Despite these developments, traditional approaches to estimating Lipschitz constants often involve solving large matrix verification problems, with computational costs that increase substantially as network depth grows (Xu and Sivaranjani 2024).

Our work diverges from these traditional approaches by avoiding the direct computation of Lipschitz constants. Instead, we utilize fine-grained preference feedback as an implicit Lipschitz regularizer in CTR models for improved robustness and generalizability. The implicity of the Lipschitz regularizer flexibly adapts the model learning such that the smoothness of its output converges to a degree that fits specifically to the properties of trained data.