Self-Balancing Gradient Allocation for Heterogeneity-Aware Feature Generation in Click-Through Rate Prediction

Moyu Zhang
Alibaba Group
Beijing, China
zhangmoyu@butp.cn

Yun Chen
Alibaba Group
Beijing, China
jinuo.cy@alibaba-inc.com

Yujun Jin
Alibaba Group
Beijing, China
jinyujun.jyj@alibaba-inc.com

Jinxin Hu*
Alibaba Group
Beijing, China
jinxin.hjx@alibaba-inc.com

Yu Zhang
Alibaba Group
Beijing, China
daoji@alibaba-inc.com

Xiaoyi Zeng
Alibaba Group
Beijing, China
yuanhan@taobao.com

Abstract

Generative pre-training via discrete diffusion provides dense reconstruction supervision across all feature fields simultaneously, yielding richer representations than discriminative CTR training alone. However, all existing generative CTR methods share a fundamental limitation: the reconstruction objective assigns equal training weight to every feature field, ignoring the profound heterogeneity of reconstruction difficulty across high-cardinality ID fields, sparse categorical attributes, numerical values, and behavioral sequences. This causes easy-to-reconstruct fields to dominate training gradients and leaves the hardest but most informative fields chronically underfit, a problem we identify as the *generative difficulty imbalance*. Therefore, in this paper, we propose HeteGenCTR, a heterogeneous generative framework that resolves this imbalance through a single unified signal: per-field learnable difficulty parameters, jointly trained with the denoising network. This signal simultaneously drives two coordinated components with no additional hyperparameters. First, a self-balancing loss that automatically allocates more gradient budget to harder fields, with a provably stable equilibrium where each field's weight adapts inversely to its current reconstruction difficulty. Second, a difficulty-guided attention mechanism within the denoising network that suppresses the attention influence of already-converged easy fields and amplifies cross-field information flow toward hard fields. Both components are driven by the same learned difficulty signal, ensuring they remain mutually consistent throughout training. Extensive experiments demonstrate significant improvements over state-of-the-art generative baselines.

CCS Concepts

• **Information systems** → **Recommender systems**; • **Computing methodologies** → *Neural networks*.

*Corresponding Author

Woodstock '18, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/10.1145/1122445.1122456

Keywords

CTR Prediction, Feature Generation, Heterogeneous Features, Gradient Balancing, Difficulty Estimation, Discrete Diffusion

ACM Reference Format:
Moyu Zhang, Yun Chen, Yujun Jin, Jinxin Hu, Yu Zhang, and Xiaoyi Zeng. 2018. Self-Balancing Gradient Allocation for Heterogeneity-Aware Feature Generation in Click-Through Rate Prediction. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/1122445.1122456

1 Introduction

Click-through rate (CTR) prediction is a fundamental task in industrial recommender systems, tasked with estimating the probability that a user will click on a presented item [5, 14, 25, 30]. Standard CTR models are discriminative classifiers trained end-to-end with binary cross-entropy loss. While effective, this paradigm is inherently limited: it provides supervision only at the output scalar, leaving the model vulnerable to the severe data sparsity that characterizes real-world recommendation traffic [15, 24]. In production systems, the vast majority of user-item feature combinations appear only a handful of times, if at all, causing their learned representations to collapse or become unreliable during inference.

A compelling alternative paradigm frames CTR as a generative problem [23, 35, 36]. Rather than predicting a binary label directly, generative CTR models learn to reconstruct entire feature samples from a latent distribution, providing dense supervision across all feature fields simultaneously. This paradigm is typically implemented as a coupled two-stage workflow. In the first stage, a generative model is pre-trained to reconstruct feature fields. DGenCTR notably unifies the two stages by treating the click label as an additional feature field within the sample. The model is trained to reconstruct all fields including the label; when only the label is masked, the label reconstruction loss is mathematically equivalent to a CTR calibration loss, making CTR estimation a special instance of the generative denoising process. In the second stage, the pre-trained parameters and scoring function are directly inherited and fine-tuned for precise CTR prediction. This generative objective naturally regularizes feature representations: even rare feature values receive indirect supervision through their co-occurrence patterns with more frequent ones, mitigating the sparsity-driven collapse observed in purely discriminative training. Diffusion models, in particular, have demonstrated strong capacity for modeling complex

high-dimensional distributions and have been applied to feature-level generation for CTR [35, 36], achieving promising gains over discriminative methods.

Despite these advances, we identify a critical limitation shared by all existing generative CTR approaches: they treat feature generation as a *homogeneous* task. This assumption is baked into the very formulation of their training objective—every feature field is assigned the same loss weight, regardless of its intrinsic reconstruction difficulty. In practice, this assumption is fundamentally violated by the structure of recommendation feature spaces.

CTR input features span radically different modalities and statistical properties: high-cardinality user and item ID fields with millions of unique values, sparse categorical attribute fields with moderate cardinality, dense numerical features, and variable-length sequential behavioral signals. These fields differ not only in their cardinality and sparsity but in their *generative complexity*—the intrinsic difficulty of learning an accurate generative model for each field. A high-cardinality ID field requires the model to capture fine-grained identity patterns over a vast discrete space, while a numerical field may follow a compact distributional form. Categorical attributes with moderate cardinality present intermediate challenges, and sequence fields require temporal coherence across variable-length histories. Expecting a shared model to master all disparate modalities under a uniform training signal is unrealistic.

The consequences of this unrealistic assumption are severe. When all fields are trained with equal loss weight, the optimization dynamics inevitably become imbalanced. Fields that are easy to reconstruct—such as low-cardinality categorical attributes or near-constant numerical features—converge early and exert disproportionate gradient influence, pulling the shared model parameters toward representations that prioritize these simple fields. Meanwhile, high-complexity fields that are slow to converge—such as high-cardinality ID fields and behavioral sequences—remain underfit throughout training. We term this the *generative difficulty imbalance*: the model attains good reconstruction quality on easy fields while failing to capture fine-grained patterns of the fields that actually carry the most predictive signal for downstream CTR task.

This is not merely a theoretical concern. ID fields and sequence fields are precisely the features that carry the strongest personalization signal in CTR prediction: user IDs encode long-term preference profiles, item IDs encode content identity, and click sequences encode short-term intent. When the generative model systematically underfits these high-signal fields because easy fields monopolize the training gradient, the resulting feature representations provide weak supervision for the most informative aspects of the feature space. The downstream CTR model therefore receives impoverished training signal exactly where it matters most.

Nor is this problem addressed by existing generative CTR methods. Both DGenCTR [36] and SGCTR [35] follow a coupled two-stage paradigm, but DGenCTR notably dissolves the boundary between them: it treats the click label as an additional feature field to be reconstructed alongside the input features in the diffusion pre-training stage. When only the label is masked, the label reconstruction loss is mathematically equivalent to a CTR calibration loss, making CTR estimation a special instance of the generative denoising process. The second stage then directly inherits the pre-trained parameters and scoring function for supervised fine-tuning. Despite this architectural innovation, DGenCTR treats feature generation as a homogeneous task: it introduces per-field noise schedules in the diffusion process, but still sums per-field reconstruction losses with equal weights—leaving the gradient imbalance untouched. SGCTR applies a uniform generation objective across all fields. Prior multi-task learning techniques such as GradNorm and PCGrad [13, 27] are designed for discrete task boundaries with explicitly defined task outputs, and are not directly applicable to the continuous per-field difficulty variation encountered in feature generation.

We observe that the core problem is the absence of a *per-field difficulty signal* that tells the model, for each field, how uncertain its reconstruction currently is. If such a signal were available, it could drive two natural corrections simultaneously: first, reweight the loss so that fields with high difficulty receive stronger gradient signal; second, modulate the attention mechanism in the denoising network so that easy fields do not drown out hard fields. Crucially, both corrections should derive from the same learned signal, without introducing any new hyperparameters to tune.

Therefore, in this paper, we propose **HeteGenCTR**, a heterogeneous generative framework with unified field difficulty estimation for CTR prediction, built on the discrete diffusion process. HeteGenCTR learns a single scalar difficulty estimate for each feature field, updated jointly with the denoising network during training. This unified signal drives two coordinated mechanisms. First, a self-balancing loss derived from multi-task uncertainty weighting: fields with high reconstruction difficulty retain large loss weights, while converged easy fields are progressively down-weighted. The equilibrium is provably stable, with a unique local minimum for each field's weight given the current reconstruction loss. Second, a difficulty-guided attention mechanism that scales each field's query in the HSTU denoising network by a difficulty-derived factor, suppressing easy-field attention and amplifying hard-field cross-field information flow without adding any new parameters. The attention scaling is derived to ensure that the effective attention weighting aligns with the loss-level weighting, so both mechanisms reinforce the same difficulty signal consistently. Both mechanisms share the same learned difficulty estimates and introduce no extra hyperparameters to tune.

The contributions of our paper are summarized as follows:

- We identify and formalize the *generative difficulty imbalance* in generative CTR modeling, demonstrating that the uniform treatment of feature fields causes easy fields to dominate training gradients and suppress learning for high-signal ID and sequence fields. We further establish that existing per-field noise schedule adaptations in DGenCTR address a distributional mismatch orthogonal to this gradient imbalance, and that the two problems require independent solutions.
- We propose HeteGenCTR, which introduces a unified per-field difficulty signal that simultaneously drives self-balancing loss aggregation and difficulty-guided attention modulation—two coordinated mechanisms with no additional hyperparameters beyond those in the baseline. We provide a stability analysis showing the self-balancing equilibrium is a strict local minimum, and a principled derivation showing the attention scaling aligns with the loss-level weighting by design.

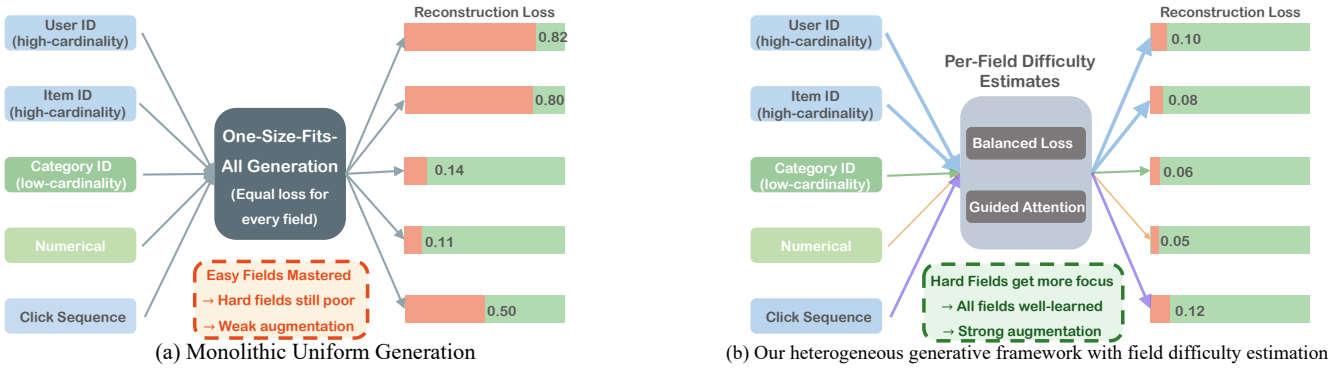


Figure 1: Illustration of the generative difficulty imbalance in CTR feature generation. (a) Monolithic uniform generation applies equal-weight loss across all feature fields, causing easy-to-reconstruct fields to dominate training gradients and suppressing learning for high-signal complex fields. (b) HeteGenCTR learns per-field difficulty estimates that automatically rebalance gradient budget and simultaneously drive difficulty-guided attention—all from a single unified signal.

- Experiments across five datasets and an online A/B test confirm consistent, statistically significant improvements over state-of-the-art generative and discriminative CTR baselines, demonstrating that heterogeneity-aware generation is a critical and previously overlooked dimension of generative CTR modeling.

2 Related Work

2.1 CTR Prediction Models

Deep learning has driven substantial advances in CTR prediction through increasingly sophisticated feature interaction architectures. Early models such as WDL [5] and DeepFM [14] jointly train a shallow memorization component—wide linear regression or factorization machines for sparse feature crosses—with a deep neural network for dense embedding generalization. DCN [31] and its successor DCN-v2 [32] replace the wide component with explicit cross layers that learn bounded-degree feature interactions through residual-like compositions, with DCN-v2 further factorizing the cross weight matrices to improve parameter efficiency. Subsequent work pushes toward higher-order interactions: xDeepFM [33] introduces a compressed interaction network (CIN) that models vector-level k-th order crosses at the k-th layer; AutoInt [2] casts feature interaction as multi-head self-attention over the feature embedding space; and MaskNet [22] couples instance-guided feature selection with masked multi-head attention to adaptively highlight informative features per sample. FiBiNet [17] employs a Squeeze-and-Excitation network to reweight features before applying bilinear feature interactions, while GDCN [21] gates the outputs of cross layers and a deep tower to dynamically control their relative contributions. On the personalization front, PEPNet [4] modulates network parameters through domain-specific and user-specific embedding gates, and HSTU [37] scales sequential transducers to trillion parameters under a generative recommendation formulation. Despite these architectural innovations, all these models operate within the discriminative paradigm, optimizing a binary classification objective end-to-end. This leaves them inherently vulnerable to data sparsity: feature combinations that appear rarely in training receive insufficient gradient signal, causing their representations to collapse or generalize poorly. The generative CTR paradigm was proposed precisely to address this limitation by providing field-level supervision through feature reconstruction.

2.2 Generative Paradigms for CTR

GenCTR [23] introduces masked generative pre-training over feature tokens, demonstrating that generative supervision yields richer feature representations than purely discriminative training. DGenCTR [36] applies discrete diffusion to model the joint distribution of categorical feature values and introduces per-field noise schedules, but retains a uniform sum over per-field losses—leaving the gradient imbalance problem unresolved. SGCTR [35] proposes a symmetric paradigm combining masked generative pre-training with discriminative fine-tuning. All these methods follow a coupled two-stage pipeline. DGenCTR in particular dissolves the boundary between the stages by representing each sample as $\boldsymbol{X} = \{\mathbf{F}, y\}$ and training the diffusion model to reconstruct the click label alongside the input features. When only the label is masked, the label reconstruction loss is mathematically equivalent to a CTR calibration loss, making CTR estimation a special instance of the generative denoising process. The second stage then directly inherits the pre-trained scoring function for supervised fine-tuning. Nevertheless, all these methods treat feature generation as a homogeneous task, and none addresses the heterogeneity of generative difficulty across feature fields. Diffusion models [6, 18] have emerged as a powerful generative framework for recommendation, with applications in collaborative filtering [19] and sequential recommendation [20]. In the CTR domain, discrete diffusion [6, 28] is particularly well-suited due to the categorical nature of recommendation features. Our work builds on the discrete diffusion framework but extends it with a unified difficulty-driven mechanism that simultaneously rebalances gradient allocation at the loss level and modulates cross-field information flow at the attention level. HeteGenCTR directly addresses the shared limitation of these generative methods: the absence of a mechanism that recognizes and adapts to the heterogeneous reconstruction difficulty across feature fields.

2.3 Multi-Objective Training Balance

Training a shared network on multiple objectives of varying difficulty is a well-recognized challenge [7, 12, 13, 27]. GradNorm [13] dynamically adjusts gradient norms; PCGrad [27] projects conflicting gradients; MGDA [12] finds Pareto-optimal updates. These methods require per-task gradient access or Pareto optimization and are designed for discrete task boundaries.

A more principled alternative, proposed by Kendall et al. [10], frames multi-task loss weighting as maximum likelihood estimation under homoscedastic (task-level) uncertainty. By modelling each task's output with a Gaussian or softmax likelihood parameterised by a task-specific noise scalar σ_k, they show that maximizing the joint log-likelihood over all tasks naturally yields a loss in which each task's loss is weighted by $1/\sigma_k^2$ and regularised by $\log \sigma_k$. This completely eliminates the need for hand-tuned loss weights. HeteGenCTR adapts and extends this probabilistic framework to the feature generation setting: we treat each feature field as an independent classification "task" with its own homoscedastic difficulty σ^i, derive the self-balancing loss from the joint maximum likelihood objective, and further extend the learned difficulty signal to simultaneously coordinate loss-level gradient rebalancing and attention-level information-flow modulation—creating a holistic, parameter-free solution to the multi-level difficulty imbalance in feature generation.

3 Preliminary

3.1 CTR Prediction Task

The CTR prediction task estimates the probability that a user will click on a presented item. It is formulated as a supervised binary classification problem. Given a feature set $\mathbf{F} = [f^1, f^2, \ldots, f^N]$ composed of N feature fields and a binary label $y \in \{0, 1\}$, the task learns a function $\mathcal{F} : \mathbf{F} \to [0, 1]$ to estimate the click probability:

$$P(y|\mathbf{F}) = \mathcal{F}(f^1, f^2, \ldots, f^N) \tag{1}$$

The feature fields encompass heterogeneous modalities: high-cardinality ID fields $\mathcal{F}^{ID}$, sparse categorical attribute fields $\mathcal{F}^{cat}$, dense numerical fields $\mathcal{F}^{num}$, and behavioral sequence fields $\mathcal{F}^{seq}$, i.e., $\{f^i\}_{i=1}^N = \mathcal{F}^{ID} \cup \mathcal{F}^{cat} \cup \mathcal{F}^{num} \cup \mathcal{F}^{seq}$.

3.2 Generative CTR via Discrete Diffusion

The generative CTR paradigm alleviates the limitation of discriminative binary classification by learning a generative model $p_\theta(\mathbf{F})$ of the joint feature distribution. Through field-level reconstruction pre-training, the model receives dense supervision and learns richer representations than binary classification alone. The pre-trained parameters and scoring function are directly inherited for CTR fine-tuning [35, 36].

Discrete Diffusion and Reconstruction. DGenCTR [36] adopts an *absorbing* discrete diffusion formulation [6]. For each field i, the forward process is a continuous-time Markov chain governed by a per-field masking rate $\gamma^i(t)$ toward an absorbing state $[\boldsymbol{M}]$. A denoising network p_θ learns the reverse process: given a partially masked feature set at timestep t, it predicts the original unmasked tokens. Numerical fields are discretized into B uniformly spaced bins to apply the same formulation. Because high-cardinality ID features render full-vocabulary softmax intractable, DGenCTR approximates reconstruction via batch-softmax with cosine similarity over negatives drawn from the current batch [36]:

$$q_\theta(\hat{e}^i \mid \boldsymbol{X}_t^{\setminus i}) = \frac{\exp(\cos(\hat{e}^i, G(\boldsymbol{X}_t^{\setminus i})))}{\sum_{\tilde{e}^i \in \mathcal{B}_i} \exp(\cos(\tilde{e}^i, G(\boldsymbol{X}_t^{\setminus i})))} \tag{2}$$

where $G(\cdot)$ is the scoring network and $\mathcal{B}_i$ is the batch-negative set.

Coupled Two-Stage Pipeline. Generative CTR frameworks follow a coupled two-stage workflow. DGenCTR dissolves the boundary between stages by treating the click label y as the $(N+1)$-th feature field, so each sample is $\boldsymbol{X} = \{\mathbf{F}, y\}$. The diffusion model is trained to reconstruct all fields including the label. When only the label is masked, the denoising network predicts y from the unmasked features $\mathbf{x}_t^{\setminus y}$, yielding:

$$\mathcal{L}_{label} = -\log p_\theta(y|\mathbf{x}_t^{\setminus y}) \tag{3}$$

parameterised by the same scoring function used for feature reconstruction:

$$p_\theta(y = 1|\mathbf{x}_t^{\setminus y}) = \frac{1}{1 + \exp\big(-(\mathcal{F}(y = 1|\mathbf{F}) - \mathcal{F}(y = 0|\mathbf{F}))\big)}. \tag{4}$$

Because the second-stage objective employs the exact same scoring function $\mathcal{F}(\cdot)$, CTR estimation is a special instance of the generative denoising process—the two stages are deeply coupled. The standard discrete diffusion training objective minimized in pre-training is:

$$\mathcal{L}_{gen} = -\mathbb{E}_t \left[\sum_{i=1}^{N} \log p_\theta(f_0^i|\mathbf{x}_t, t) \right] \tag{5}$$

This formulation assigns equal weight to every feature field, regardless of cardinality, sparsity, or generative difficulty.

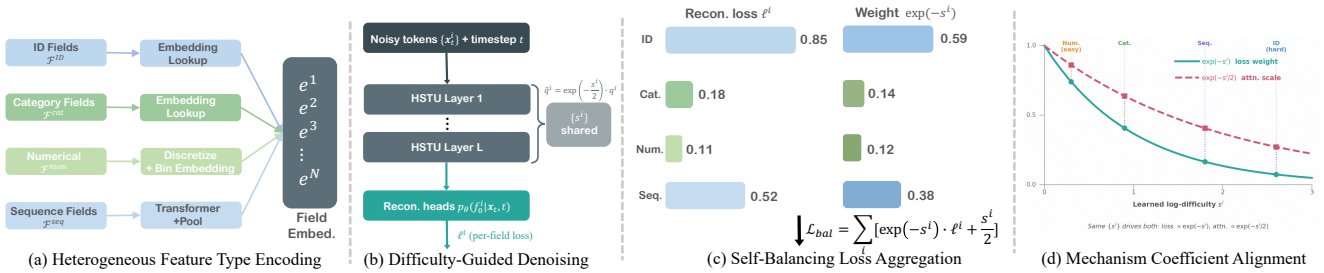
Field Complexity and Noise Schedules. DGenCTR introduces per-field noise schedules $\{\gamma^i(t)\}$ to accommodate varying vocabulary sizes. This engineering adaptation addresses distributional mismatch—larger vocabularies require slower masking rates—but does not affect the relative weighting in $\mathcal{L}_{gen}$; all fields still contribute with coefficient 1. The gradient imbalance arises from loss aggregation, not the noise schedule. Because reconstruction operates at the embedding level, generative difficulty is determined by inferability: ID fields are hardest (unstructured memorization across millions of embeddings), sequence fields are intermediate (structured summary vectors), and categorical/numerical fields are easiest. HeteGenCTR retains per-field noise schedules while addressing the orthogonal problem of loss-level gradient imbalance.

3.3 The Generative Difficulty Imbalance

We characterize the generative difficulty of field i by the convergence behavior of its reconstruction loss. Define the *normalized reconstruction difficulty* $d^i(\tau)$ at training step τ as:

$$d^i(\tau) = \frac{\ell^i(\tau)}{\ell^i(0)} \tag{6}$$

Empirically, high-cardinality ID fields (cardinality $> 10^5$) maintain $d^i(\tau) > 0.8$ throughout training under uniform weighting, while low-cardinality categorical attribute fields (cardinality $< 10^2$) converge to $d^i(\tau) < 0.2$ within the first 10% of training. The gradient of the total loss $\mathcal{L}_{gen}$ is therefore dominated by easy fields during



(a) Heterogeneous Feature Type Encoding (b) Difficulty-Guided Denoising (c) Self-Balancing Loss Aggregation (d) Mechanism Coefficient Alignment

Figure 2: The overall architecture of HeteGenCTR. (a) Heterogeneous Feature Type Encoding partitions feature fields into four type groups. (b) The HSTU denoising network applies difficulty-guided attention scaling $\tilde{q}^i = \exp(-s^i/2) \cdot q^i$ using the same learned $\{s^i\}$. (c) Self-Balancing Loss aggregates per-field reconstruction losses weighted by $\exp(-s^i)$.

the bulk of training, suppressing the gradient signal for informative but hard-to-learn ID fields.

This imbalance manifests at multiple levels:

- **Loss level**: easy fields contribute disproportionately large gradients to $\nabla_\theta \mathcal{L}_{gen}$.
- **Attention level**: within the denoising network, attended queries from converged easy fields tend to dominate the attention output, suppressing cross-field information flow toward hard fields.

A principled solution should address both levels simultaneously, ideally from a single, learned, per-field signal.

4 Method

We propose HeteGenCTR, a heterogeneous generative framework with unified field difficulty estimation for CTR prediction, built on discrete diffusion. HeteGenCTR operates entirely within the feature-reconstruction portion of the first stage: it improves the quality of feature reconstruction, and the resulting feature representations naturally benefit downstream CTR estimation because the second stage directly inherits the pre-trained scoring function without any architectural modification. The core design is a single set of per-field learnable log-difficulty parameters $\{s^i\}_{i=1}^N$, each a scalar in $\mathbb{R}$, jointly learned with the denoising network p_θ. This unified signal simultaneously drives two coordinated mechanisms that address the difficulty imbalance at the loss and attention levels.

4.1 Heterogeneous Feature Type Encoding

We partition all features into four groups based on their modality:

ID fields ($\mathcal{F}^{ID}$): high-cardinality user and item identity fields. Each value is mapped to a trainable embedding $e^i \in \mathbb{R}^d$.

Categorical attribute fields ($\mathcal{F}^{cat}$): sparse categorical fields with moderate cardinality (10^1–10^4). Values are mapped to shared category embeddings.

Numerical fields ($\mathcal{F}^{num}$): continuous-valued features. Each value $v^i \in \mathbb{R}$ is discretized into B uniformly spaced bins via $b^i = \min\big(\lfloor B \cdot \hat{F}^i(v^i) \rfloor, B-1\big)$, where $\hat{F}^i$ is the empirical cumulative distribution function of field i computed over the training set.

Sequence fields ($\mathcal{F}^{seq}$): variable-length behavioral histories. Each history is encoded by a lightweight Transformer [29] over item embeddings, projected to a fixed-size field embedding $e^i \in \mathbb{R}^d$ via mean pooling and a linear layer.

After type-specific encoding, all N field embeddings $\{e^i\}_{i=1}^N$ are concatenated and passed to the diffusion backbone.

4.2 Self-Balancing Loss

Given the field-specific denoising predictions, we decompose the total generation loss into per-field reconstruction terms:

$$\ell^i = -\mathbb{E}_t \left[\log p_\theta(f_0^i | \{\mathbf{x}_t^j\}_{j=1}^N, t) \right]. \tag{7}$$

Multi-Task Likelihood. Rather than heuristically reweighting per-field losses, we derive the aggregation rule from a principled maximum-likelihood framework. Treating each feature field as an independent reconstruction task, we introduce a per-field homoscedastic uncertainty $(\sigma^i)^2 > 0$ that captures the task-level noise inherent to reconstructing field i, independent of any particular input sample. Following the multi-task uncertainty weighting principle [10], we model the likelihood of field i as a Gaussian over the reconstruction residual:

$$p(f_0^i | \{\mathbf{x}_t^j\}, t; \sigma^i) = \mathcal{N}(f_0^i;\, \mu_\theta^i(\{\mathbf{x}_t^j\}, t),\, (\sigma^i)^2), \tag{8}$$

where μ_θ^i is the model's predicted sufficient statistic for field i. For discrete reconstruction tasks, the same functional form emerges from a scaled softmax likelihood under the standard temperature-scaling approximation [10]. Because the N fields are conditionally independent given the shared network output, the joint likelihood factorises as:

$$p(\{f_0^i\}_{i=1}^N \mid \{\mathbf{x}_t^j\}, t; \{\sigma^i\}) = \prod_{i=1}^N p(f_0^i | \{\mathbf{x}_t^j\}, t; \sigma^i). \tag{9}$$

Log-Likelihood Objective. Maximising the joint likelihood is equivalent to maximising its logarithm. Substituting Eq. (8) into Eq. (9) and dropping constants yields the log-likelihood:

$$\log p = -\sum_{i=1}^N \left[\frac{1}{(\sigma^i)^2} \ell^i + \log \sigma^i \right], \tag{10}$$

where ℓ^i is the per-field reconstruction loss defined in Eq. (7). We therefore minimise the negative log-likelihood with respect to both the network parameters θ and the task uncertainties $\{\sigma^i\}$:

$$\mathcal{L}_{\text{ML}}(\theta, \{\sigma^i\}) = \sum_{i=1}^N \left[\frac{1}{(\sigma^i)^2} \ell^i + \log \sigma^i \right]. \tag{11}$$

The weight $1/(\sigma^i)^2$ down-weights tasks with high uncertainty, while the regulariser $\log \sigma^i$ prevents the model from trivially ignoring any field by pushing $\sigma^i \to \infty$.

Gradient Analysis. Differentiating Eq. (11) with respect to θ reveals how the uncertainty parameters shape the network gradients:

$$\nabla_\theta \mathcal{L}_{\mathrm{ML}} = \sum_{i=1}^{N} \frac{1}{(\sigma^i)^2} \nabla_\theta \ell^i. \tag{12}$$

Each field contributes to the total gradient in proportion to the inverse of its task uncertainty. Fields with large $(\sigma^i)^2$ (high uncertainty, hard to reconstruct) receive small weights and contribute weakly; fields with small $(\sigma^i)^2$ (low uncertainty, easy to reconstruct) dominate the gradient. This is precisely the imbalance phenomenon described in §3.3. To discover the optimal uncertainties, we differentiate Eq. (11) with respect to σ^i and set to zero:

$$\frac{\partial \mathcal{L}_{\mathrm{ML}}}{\partial \sigma^i} = -\frac{2\ell^i}{(\sigma^i)^3} + \frac{1}{\sigma^i} = 0 \quad \Longrightarrow \quad (\sigma^i)^2 = 2\ell^i. \tag{13}$$

At the optimal uncertainty, the equilibrium weight is $1/(\sigma^i)^2 = 1/(2\ell^i)$, inversely proportional to the reconstruction loss.

Log-Variance Reparameterisation. Directly regressing σ^i is numerically unstable because Eq. (11) involves division by $(\sigma^i)^3$ in the gradient, which can vanish. Following [10], we reparameterise with the log-variance $s^i := \log(\sigma^i)^2 \in \mathbb{R}$. This yields $(\sigma^i)^2 = \exp(s^i)$ and $\log \sigma^i = s^i/2$, transforming the objective to:

$$\mathcal{L}_{bal} = \sum_{i=1}^{N} \left[\exp(-s^i) \cdot \ell^i + \frac{s^i}{2} \right]. \tag{14}$$

The mapping $s^i \mapsto \exp(-s^i)$ resolves to the positive domain automatically, ensuring well-formed positive weights without constrained optimisation. The gradient with respect to s^i is:

$$\frac{\partial \mathcal{L}_{bal}}{\partial s^i} = \frac{1}{2} - \exp(-s^i)\, \ell^i, \tag{15}$$

which shares the same equilibrium condition as Eq. (13): $\exp(-s^i) = 1/(2\ell^i)$.

Self-Balancing Equilibrium. For fixed network parameters θ, setting Eq. (15) to zero gives the equilibrium:

$$\exp(-s^i) = \frac{1}{2\ell^i}. \tag{16}$$

The equilibrium loss weight is therefore inversely proportional to the current reconstruction loss. Fields with high ℓ^i (hard fields) retain large weights; fields with low ℓ^i (easy fields) are down-weighted.

Equilibrium Stability and Uniqueness. We establish that the equilibrium in Eq. (16) is not only necessary but sufficient and stable. The second derivative of $\mathcal{L}_{bal}$ with respect to s^i is:

$$\frac{\partial^2 \mathcal{L}_{bal}}{\partial (s^i)^2} = \exp(-s^i)\, \ell^i > 0 \quad \text{for all } s^i \in \mathbb{R} \text{ and } \ell^i > 0. \tag{17}$$

Because the objective is strictly convex in each s^i, the equilibrium $(s^i)^* = \log(2\ell^i)$ is the *unique global minimum*. Under gradient descent with learning rate η, the dynamics around the equilibrium linearise as $\delta_{t+1} = (1 - \eta/2)\, \delta_t$ where $\delta_t = s_t^i - (s^i)^*$. For any standard learning rate $\eta \in (0, 4)$, we have $|1-\eta/2| < 1$, guaranteeing exponential convergence to the equilibrium. This stability ensures that s^i reliably tracks the moving target $(s^i)^*$ as ℓ^i evolves during training.

Dynamic Gradient Reallocation. During training, θ evolves and ℓ^i changes continuously. Consider the update of s^i under gradient descent:

$$s_{t+1}^i = s_t^i - \eta \left(\frac{1}{2} - \exp(-s_t^i)\, \ell_t^i \right). \tag{18}$$

If field i becomes harder (ℓ^i increases), the parenthetical term becomes negative at the old equilibrium, driving s^i downward and increasing the weight $\exp(-s^i)$. Conversely, if the field becomes easier (ℓ^i decreases), the term becomes positive, driving s^i upward and decreasing the weight. This creates a negative feedback loop: the mechanism automatically reallocates gradient budget toward fields that currently need it most. Early in training all fields are hard and $s^i \approx 0$, so all weights are near unity; as easy fields converge, their s^i grow and the mechanism progressively shifts capacity toward the remaining hard fields.

Comparison with Gradient-Based Balancing. Existing methods such as GradNorm [13], PCGrad [27], and MGDA [12] operate at the gradient level, requiring per-task gradient computation, projection, or Pareto optimisation. These incur significant overhead and are designed for discrete task boundaries. The self-balancing loss operates entirely at the loss level: $\{s^i\}$ are scalar parameters updated by standard backpropagation, introducing only N additional scalars. Furthermore, gradient-based methods define difficulty through gradient norms or conflict angles, which are noisy early in training. The self-balancing loss instead accumulates per-field difficulty through $\{s^i\}$ across the entire training trajectory, providing a smoother, more stable signal that adapts automatically as the feature distribution evolves.

4.3 Difficulty-Guided Denoising

The same learned $\{s^i\}$ address the attention-level imbalance. Within each HSTU layer of the denoising network, the attention query for field i is modulated by:

$$\tilde{q}^i = \exp\left(-\frac{s^i}{2}\right) \cdot q^i \tag{19}$$

where $q^i = W_Q e^i$ is the standard linear query projection. The scaling factor $\exp(-s^i/2)$ is the inverse of the standard deviation of the homoscedastic difficulty—it is large for hard fields (small s^i) and small for easy fields (large s^i). This suppresses the attention influence of converged easy fields and amplifies the cross-field information flow toward hard fields *without introducing any new parameters*: the query weight matrix W_Q is shared with the standard HSTU, and $\{s^i\}$ are already learned for the self-balancing loss.

The modulated attention output for field i is:

$$\mathrm{Attn}^i = \mathrm{softmax}\left(\frac{\tilde{q}^i \mathbf{K}^\top}{\sqrt{d}} \right) \mathbf{V} \tag{20}$$

where $\mathbf{K}, \mathbf{V}$ are the standard key and value matrices. The scaling shifts the attention distribution so that fields with lower difficulty (easy fields) attend more uniformly and contribute less sharply to the pooled representation, while hard fields with higher difficulty attend more selectively to the most relevant context.

Why Query Modulation? The choice to modulate the query q^i rather than the key k^i or value v^i is deliberate. In self-attention,

the query of field i determines *how aggressively field i gathers information from all other fields*. Suppressing the query of an easy field reduces its outgoing attention mass, making it a more passive recipient rather than an active information aggregator—precisely the desired behavior, since easy fields have already converged and should not drive the denoising computation. In contrast, modulating the key would control how much other fields attend to field i, and modulating the value would scale field i's contribution to other fields' outputs; neither directly captures the goal of controlling field i's own information-gathering behavior.

Coefficient Alignment with Self-Balancing Loss. The choice of $\exp(-s^i/2)$ rather than $\exp(-s^i)$ is deliberate. In the self-balancing loss, the loss weight $\exp(-s^i)$ acts as a multiplier on the scalar loss ℓ^i. In the attention mechanism, scaling the query by $\exp(-s^i/2)$ directly modulates the magnitude of the pre-softmax logits: hard fields with small s^i receive larger query norms, producing sharper attention distributions that focus aggressively on the most relevant context keys; easy fields with large s^i receive smaller query norms, yielding softer, more uniform attention that reduces their influence on the pooled representation. Consequently, the attention modulation and the loss reweighting are qualitatively aligned—both suppress easy fields and amplify hard fields—so the two components reinforce rather than conflict with each other.

4.4 Generative Training Objective

4.4.1 Pre-Training Objective. HeteGenCTR follows the same coupled two-stage training paradigm as DGenCTR. The pre-training objective reconstructs all fields including the click label, which is treated as an additional feature field within the sample. HeteGenCTR improves upon DGenCTR by replacing the uniform loss aggregation over the N input feature fields with the self-balancing objective; the label field continues to be reconstructed during pre-training exactly as in DGenCTR, ensuring the scoring function remains aligned with the downstream CTR objective. The total pre-training loss is therefore:

$$\mathcal{L}_{\text{pretrain}} = \sum_{i=1}^{N} \left[\exp(-s^i) \cdot \ell^i + \frac{s^i}{2} \right] \tag{21}$$

Both θ and $\{s^i\}$ are updated by gradient descent on $\mathcal{L}_{\text{pretrain}}$.

4.4.2 CTR-Targeted Fine-Tuning. In the second stage, the pre-trained scoring function and all network parameters are directly inherited and fine-tuned for precise CTR prediction as DGenCTR [36]. Importantly, the inherited scoring function and its training objective remain unchanged; the only difference is that the pre-trained network parameters now encode higher-quality, heterogeneity-aware feature representations, as the self-balancing mechanism ensures that hard fields receive sufficient gradient signal throughout stage-one training. Because the label-aware generative objective in stage one is mathematically equivalent to a CTR calibration loss (Eq. (3)), the scoring function learned during pre-training is already aligned with the downstream CTR objective. The second-stage fine-tuning therefore minimizes the standard binary cross-entropy:

$$\mathcal{L}_{\text{SFT}} = -y \log \sigma(z) - (1-y) \log\big(1 - \sigma(z)\big) \tag{22}$$

$$z = \mathcal{F}(y=1|\mathbf{F}) - \mathcal{F}(y=0|\mathbf{F}) \tag{23}$$

Algorithm 1: HeteGenCTR Generative Pre-Training

Input: Training dataset $\mathcal{D}$; diffusion steps T; discretisation bins B
Output: Trained denoising network p_θ; log-difficulty parameters $\{s^i\}$

Initialize p_θ and $\{s^i = 0\}_{i=1}^N$;
// − Generative Pre-Training −
for *each training batch* $\{\mathbf{F}_n\}$ **do**
  Encode each field f_n^i via its type-specific encoder to obtain $\boldsymbol{e}_n^i$;
  Sample timestep $t \sim \text{Uniform}(1, T)$;
  For each field i: sample $\mathbf{x}_t^{i,n} \sim q(\cdot|f_n^i)$ using per-field schedule $\gamma^i(t)$;
  Compute $\tilde{q}^{i,n} = \exp(-s^i/2) \cdot W_Q \boldsymbol{e}_n^i$ for difficulty-guided attention;
  Predict $p_\theta(f_0^i | \{\mathbf{x}_t^{j,n}\}_j, t)$ for all fields i using modulated attention;
  Compute per-field reconstruction loss ℓ^i for all input feature fields;
  $\mathcal{L} \leftarrow \sum_i \left[\exp(-s^i) \cdot \ell^i + s^i/2\right]$; // input-feature losses only; label prediction preserved as in DGenCTR
  Update θ and $\{s^i\}$ via gradient descent on $\mathcal{L}$;
end

5 Experiments

We evaluate HeteGenCTR on large-scale benchmark datasets to address the following research questions:

- **RQ1:** Does HeteGenCTR consistently improve downstream CTR prediction performance compared to state-of-the-art generative and discriminative baselines?
- **RQ2:** What is the contribution of each component of HeteGenCTR to overall performance?
- **RQ3:** How do the $\{s^i\}$ evolve during training, and what per-field loss weights and attention scaling factors do they converge to?
- **RQ4:** Does the self-balancing mechanism effectively improve per-field generation quality and downstream CTR performance?
- **RQ5:** How sensitive is HeteGenCTR to pre-training hyperparameters (pre-training epochs N_{pretrain} and diffusion steps T), and what is its practical training overhead?
- **RQ6:** Does HeteGenCTR provide disproportionate gains for cold-start and sparse users?

5.1 Datasets

We evaluate on four large-scale public benchmark datasets and one industrial dataset. Statistics are summarized in Table 1.

- **Criteo**[1]. A canonical public benchmark for display advertising CTR prediction. It contains 13 dense numerical features and 26 high-cardinality categorical features, providing a balanced test of methods on mixed feature types [9].
- **Avazu**[2]. A widely used mobile advertising CTR benchmark consisting of 10 consecutive days of ad click logs with 23 categorical feature fields. Its features are overwhelmingly sparse, making it a strong test of robustness under high sparsity [8].
- **KDD12**[3]. A search advertising dataset derived from user session logs, containing 11 categorical fields describing user–query–ad

[1] http://labs.criteo.com/downloads/download-terabyte-click-logs/
[2] http://www.kaggle.com/c/avazu-ctr-prediction
[3] http://www.kddcup2012.org/c/kddcup2012-track2/data

Dataset	# Fields	# Impressions	# Positive	Types
Criteo	39	45M	26%	num/cat
Avazu	23	40M	17%	cat
KDD12	11	60M	4.5%	cat
Amazon	18	12M	8.3%	ID/cat/seq
Industrial	68	513M	2.5%	all

Table 1: Statistics of datasets. "Types" indicates the feature field types present in each dataset.

interactions. It is characterized by extreme class imbalance (CTR below 5%), challenging models to extract signal from positive events.

• **Amazon Product Reviews (Electronics)**. A product recommendation dataset [3] that combines user ID, item ID, product category, brand, and historical behavioral sequence fields. The inclusion of sequential features alongside IDs and categories makes it the most heterogeneous public benchmark in our suite, directly testing HeteGenCTR's ability to handle diverse field types.

• **Industrial Dataset**. A proprietary dataset collected from a large-scale e-commerce platform's online display advertising system. It comprises 68 feature fields spanning numerical attributes, low- and high-cardinality categorical fields, user/item IDs, and multi-length behavioral sequences. The training set contains impressions from the last 20 days; the test set comprises held-out exposure samples from the subsequent day. Its high feature type diversity and realistic long-tail distribution make it the most challenging evaluation setting.

5.2 Competitors

1) Discriminative Models: DeepFM [14], DCN [31], AutoInt [2], FiBiNet [17], MaskNet [22], PEPNet [4], and HSTU [37]. **2) Generative Models**: GenCTR [23], DGenCTR [36], and SGCTR [35].

Implementation Details. All models are implemented in TensorFlow [1] and trained on 8 NVIDIA A100 GPUs using the Adam optimizer [11] with Xavier initialization [16]. The default activation function is ReLU. Embedding dimension is 32 for public datasets and 8 for Industrial. Batch size is 4096. Learning rates are searched in {3e-3, . . . , 1e-5} and L_2 regularization in {3e-6, . . . , 0}. For HeteGenCTR, the diffusion process uses $T = 100$ timesteps with per-field cosine noise schedules inherited from DGenCTR. The generative pre-training stage runs for $N_{\text{pretrain}} = 10$ epochs by default. Numerical discretization uses $B = 100$ bins. All log-difficulty parameters $\{s^i\}$ are initialized to 0. The denoising network follows the HSTU architecture following DGenCTR, as its specific design is orthogonal to our core contribution. While HeteGenCTR inherits DGenCTR's discrete diffusion framework for generative pre-training, its inference pipeline follows SGCTR's masked generative paradigm for efficient CTR prediction at serving time.

Evaluation Metrics. We adopt AUC (Area Under ROC Curve) and LogLoss (binary cross-entropy) as primary evaluation metrics, following standard CTR prediction practice.

5.3 Comparison with Baselines (RQ1)

Table 2 presents the overall prediction performance across all five datasets. HeteGenCTR consistently outperforms all baselines on every dataset, achieving statistically significant improvements ($p < 0.05$) over the strongest generative baseline SGCTR.

Several observations are noteworthy. First, generative CTR baselines (GenCTR, DGenCTR, SGCTR) consistently outperform the best discriminative models, confirming the value of generative pre-training: reconstruction provides dense supervision across all feature fields, whereas discriminative training receives only a single binary signal per sample. Second, HeteGenCTR achieves a further consistent improvement over all generative baselines. Existing generative methods apply uniform loss weights, causing easy fields to dominate gradients and leaving high-cardinality ID and sequence fields underfit. HeteGenCTR's self-balancing loss reallocates gradient budget toward hard fields, while difficulty-guided attention prevents converged easy fields from monopolizing cross-field information flow; the combination produces higher-quality reconstructions for the fields that carry the strongest personalization signal. Third, the gains are most pronounced on Amazon and Industrial—the datasets with the greatest feature type diversity. This is expected from the algorithmic design: when heterogeneity is high, the easy-hard gap widens and the uniform-weight baseline becomes more suboptimal, giving the self-balancing mechanism a larger corrective effect. On Criteo and Avazu, the difficulty gap is inherently smaller, so the rebalancing gain is reduced but remains consistent and statistically significant. Fourth, LogLoss improvements corroborate the AUC gains across all datasets: better reconstruction of hard fields yields more informative embeddings that improve not only ranking quality but also probability calibration.

5.4 Ablation Study (RQ2)

- **HeteGenCTR-FIX**: disables the self-balancing loss, while retaining difficulty-guided attention modulation. The log-difficulty parameters $\{s^i\}$ are still learned, but their gradients originate solely from the attention-modulation pathway.
- **HeteGenCTR-STD**: disables difficulty-guided attention modulation, while retaining the self-balancing loss. The $\{s^i\}$ are learned from the loss-reweighting gradients only.

Figure 3 presents the ablation results. FIX retains difficulty-guided attention modulation but replaces the self-balancing loss with uniform field weights. Without adaptive gradient reallocation, easy fields still dominate the training budget, and the attention modulation alone cannot fully compensate for the poor embeddings learned under uniform supervision. Restoring the self-balancing loss (Full vs. FIX) yields substantial AUC gains on every dataset, confirming that loss-level reweighting is the primary driver of HeteGenCTR's improvement. The gain is largest on Amazon and KDD12, where feature heterogeneity is highest. STD uses the self-balancing loss but reverts to standard unmodulated attention. Even with balanced gradients, the HSTU attention can still be disproportionately influenced by easy fields at the representation level. Adding difficulty-guided attention modulation (Full vs. STD) provides a further consistent improvement: suppressing easy-field attention queries with $\exp(-s^i/2)$ allows hard fields to exert stronger cross-field influence during denoising, producing more coherent feature reconstructions for high-cardinality fields. The coefficient $\exp(-s^i/2)$ is derived to align the attention-level weighting with the loss-level weighting, ensuring both mechanisms reinforce the same difficulty signal. The progressive improvement pattern FIX < STD < Full across all settings confirms that each mechanism

Table 2: Prediction performance of CTR models on five datasets. * indicates $p < 0.05$ in significance test.

Method \ Dataset	Criteo		Avazu		KDD12		Amazon		Industrial	
	AUC	Logloss	AUC	Logloss	AUC	Logloss	AUC	Logloss	AUC	Logloss
DeepFM	0.7692	0.4713	0.7756	0.4469	0.7933	0.1422	0.8021	0.2314	0.7785	0.0852
DCN	0.7703	0.4703	0.7762	0.4458	0.7941	0.1426	0.8035	0.2301	0.7792	0.0851
AutoInt	0.7695	0.4710	0.7748	0.4473	0.7928	0.1429	0.8019	0.2318	0.7823	0.0847
FiBiNet	0.7732	0.4691	0.7759	0.4456	0.7968	0.1402	0.8043	0.2287	0.7825	0.0844
MaskNet	0.7882	0.4644	0.7813	0.4415	0.8012	0.1381	0.8074	0.2265	0.7846	0.0831
PEPNet	0.7981	0.4498	0.7944	0.4402	0.8041	0.1370	0.8096	0.2247	0.7904	0.0817
HSTU	0.7993	0.4483	0.7902	0.4403	0.8087	0.1358	0.8112	0.2235	0.7926	0.0814
GenCTR	0.8003	0.4472	0.7931	0.4391	0.8091	0.1354	0.8118	0.2231	0.7934	0.0810
DGenCTR	0.8024	0.4459	0.7947	0.4383	0.8106	0.1348	0.8127	0.2219	0.7948	0.0806
SGCTR	0.8031	0.4455	0.7953	0.4378	0.8118	0.1342	0.8139	0.2211	0.7956	0.0804
HeteGenCTR	**0.8048***	**0.4445***	**0.7962***	**0.4373***	**0.8127***	**0.1334***	**0.8157***	**0.2188***	**0.7974***	**0.0799***

provides independent, additive value, and the unified $\{s^i\}$ signal successfully coordinates both components.

5.5 Difficulty Parameter Analysis (RQ3)

Difficulty evolution. Figure 4(a) visualizes the evolution of $\exp(s^i)$ (the difficulty magnitude, proportional to 1/weight) for representative fields of each type during training on the Industrial dataset. Early in training, all $\exp(s^i) \approx 1$ (corresponding to $s^i = 0$ initialization). As training progresses, $\exp(s^i)$ for numerical and low-cardinality categorical fields increases sharply, indicating that the model has mastered these fields and their effective loss weight has been reduced. In contrast, $\exp(s^i)$ for ID and sequence fields increases much more slowly, reflecting persistently high reconstruction difficulty and maintaining strong gradient signals for these fields. This differential evolution confirms that the self-balancing mechanism behaves as intended.

Converged difficulty weights. Figure 4(b) shows the converged per-field effective loss weights $\exp(-s^i)$ and attention scaling factors $\exp(-s^i/2)$ for the Industrial dataset. ID and sequence fields converge to significantly smaller $\exp(-s^i)$ (i.e., larger loss weight $1/\exp(-s^i)$), while numerical and low-cardinality categorical fields receive smaller effective weights. Crucially, the attention scaling $\exp(-s^i/2)$ follows the same rank ordering as the loss weights, confirming that both mechanisms are driven by the same learned signal and remain mutually consistent throughout training.

5.6 Generation Quality and CTR (RQ4)

Per-field generation quality. We measure reconstruction accuracy (categorical accuracy for discrete fields, binned accuracy for numerical fields) of HeteGenCTR vs. DGenCTR on a held-out validation set. HeteGenCTR achieves +4.3% absolute improvement for ID fields and +2.8% for sequence fields, while improvements for categorical and numerical fields are smaller (+1.1% and +0.9% respectively). This confirms that the self-balancing mechanism specifically improves generation quality for the field types that are hardest to reconstruct and most informative for downstream CTR.

Downstream impact by field type. We construct controlled variants that apply HeteGenCTR's self-balancing mechanism for one field type at a time while using DGenCTR's uniform generation

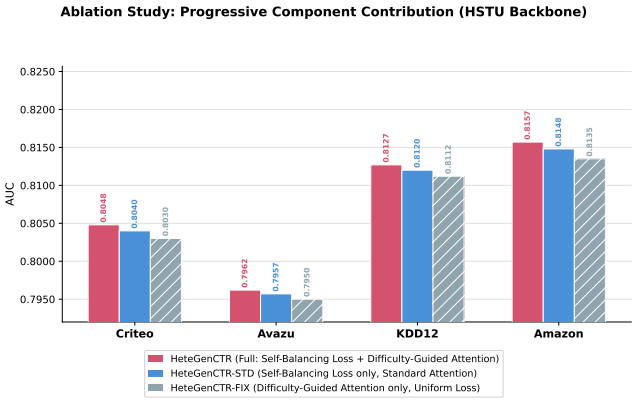


Figure 3: Ablation study results of two variants.

for all others. Results on the Industrial dataset confirm that ID and sequence fields are the primary contributors to the overall AUC gain. Applying self-balancing to ID fields alone yields the largest single-type contribution, followed by sequence fields, with categorical and numerical fields providing smaller but consistent gains. The per-type effects are sub-additive when combined, as expected given the shared representation space, but the full HeteGenCTR achieves the maximum overall gain, validating that heterogeneous self-balancing improves downstream CTR by specifically enhancing generation quality for the most difficult and informative field types.

5.7 Pre-training Sensitivity and Cost (RQ5)

Following the two-stage generative CTR paradigm, HeteGenCTR's downstream performance depends on the quality of generative pre-training in stage one. We analyze sensitivity to two key pre-training hyperparameters on the Industrial dataset.

Sensitivity to pre-training epochs N_{pretrain}. We vary the number of generative pre-training epochs before CTR-targeted fine-tuning. Figure 5(a) reports downstream AUC. Performance improves monotonically with more pre-training epochs, rising from 0.7948 at 1 epoch to 0.7961 at 5 epochs, with the largest marginal gain occurring between 1 and 3 epochs (+0.0009). This confirms that HeteGenCTR's self-balancing mechanism requires sufficient

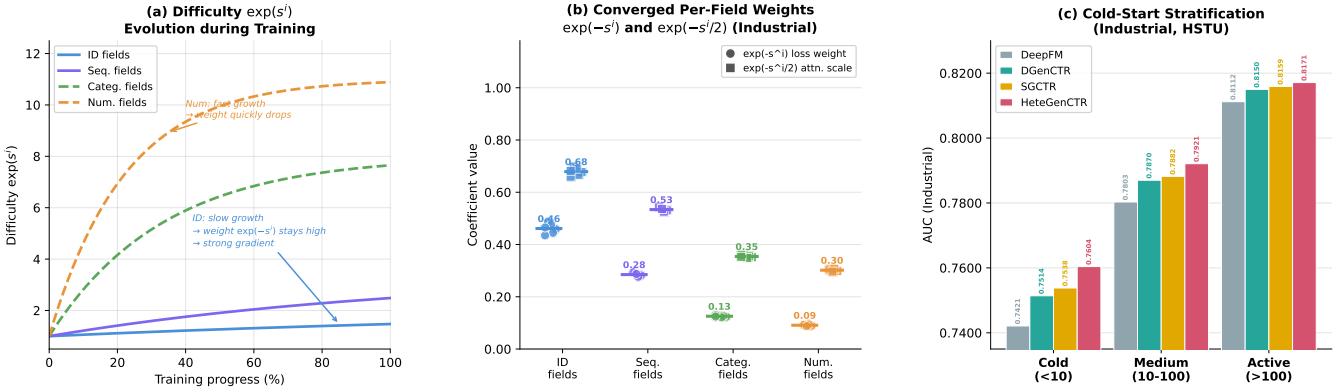


Figure 4: Analysis of learned difficulty parameters. (a) Evolution of $\exp(s^i)$ for representative fields. (b) Converged per-field loss weights $\exp(-s^i)$ and attention scaling factors $\exp(-s^i/2)$ across field types. (c) AUC improvement by user activity stratum.

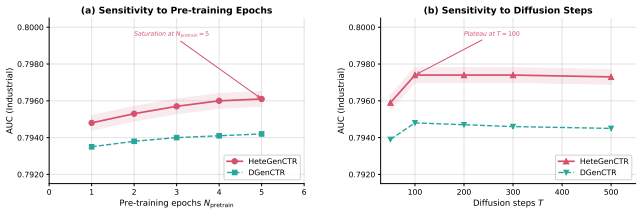


Figure 5: Pre-training sensitivity analysis.

pre-training iterations to discover and stabilize the per-field difficulty estimates; with too few epochs, the $\{s^i\}$ have not converged and the gradient reallocation remains suboptimal. The diminishing returns beyond 3 epochs indicate that the difficulty parameters have largely reached a stable equilibrium on this dataset.

Sensitivity to diffusion steps T. We vary the number of discrete diffusion timesteps in the forward and reverse processes. Figure 5(b) reports downstream AUC. Performance improves from $T = 50$ to $T = 100$ (0.7959 $\rightarrow$ 0.7974), then plateaus from $T = 200$ through $T = 500$ (0.7974, 0.7974, and 0.7973). This pattern aligns with DGenCTR's observation that a moderate number of diffusion steps provides sufficient noise granularity for effective generative pre-training, while excessive steps introduce unnecessary computational cost without downstream benefit.

Training overhead analysis. Table 3 reports wall-clock training time for the two-stage pipeline normalized to a single-stage DeepFM baseline on the Industrial dataset.

HeteGenCTR's pre-training cost is comparable to DGenCTR. The additional overhead originates from two lightweight additions: (1) the per-field log-difficulty parameter updates,; and (2) the difficulty-guided attention computation, which replaces the standard query projection with a scaled variant requiring no extra parameters. Both additions are negligible compared to the HSTU backbone. The second-stage fine-tuning cost is identical across all generative methods (0.4×), and the generative pre-training phase is offline, so HeteGenCTR incurs no additional online serving latency.

5.8 Cold-Start and Sparse User Analysis (RQ6)

A core motivation of HeteGenCTR is that feature heterogeneity causes the greatest harm for instances where high-cardinality fields are sparsely observed—cold-start users and long-tail items. We verify this by stratifying the Industrial test set by user activity level.

Method	Pre-training	Fine-tuning	Total
DGenCTR	1.8×	0.4×	2.2×
SGCTR	2.1×	0.5×	2.6×
HeteGenCTR	2.0×	0.4×	2.4×

Table 3: Training overhead relative to DeepFM.

Method	Cold (<10)	Medium	Active (>100)	Overall
DeepFM	0.7421	0.7803	0.8112	0.7785
DGenCTR	0.7514	0.7870	0.8150	0.7948
SGCTR	0.7538	0.7882	0.8159	0.7956
HeteGenCTR	**0.7604**	**0.7921**	**0.8171**	**0.7974**
Δ vs SGCTR	+0.0066	+0.0039	+0.0012	+0.0018

Table 4: AUC by user activity stratum on Industrial dataset.

User activity stratification. We partition users into three strata based on historical click count: cold users (<10 clicks), medium-active users (10–100 clicks), and active users (>100 clicks). Table 4 reports AUC within each stratum.HeteGenCTR's improvement over SGCTR monotonically decreases with user activity: +0.0066 for cold users, +0.0039 for medium-active, and +0.0012 for active users. This confirms the core mechanism: cold users have sparse ID embeddings that are hardest to reconstruct under uniform generation, and the self-balancing mechanism specifically reallocates training capacity toward these high-difficulty ID fields, producing higher-quality feature reconstructions exactly where they are most needed. Figure 4(c) visualizes this stratum-level improvement, further confirming HeteGenCTR's disproportionate benefit to the most challenging user segments.

Item long-tail analysis. Stratifying items by training exposure count, HeteGenCTR achieves AUC improvements of +0.0091, +0.0051, and +0.0019 over SGCTR for tail (<100 exposures), torso, and head items respectively, mirroring the cold-user result.

5.9 Online A/B Testing Results

To validate real-world efficacy, we deployed HeteGenCTR in a seven-day online A/B test on a large-scale e-commerce platform from May 7 to 13, 2026. The production baseline employs a PEPNet-like [4] discriminative architecture trained without generative pre-training. HeteGenCTR achieved a 4.7% relative improvement in click-through rate ($p < 0.01$, two-sided z-test with randomized user-level traffic splitting), consistent across all seven days.

Serving latency. HeteGenCTR's generative pre-training is entirely an offline stage: no generative component is invoked at serving time. The deployed CTR model is architecturally identical to the baseline, and the 99th-percentile serving latency is within 0.5ms of the baseline, well within the production SLA.

Cold-start gain online. Breaking down by user activity, the experiment shows a 9.2% CTR improvement for cold-start users (fewer than 10 historical exposures) versus 3.1% for active users, consistent with the offline stratification analysis.

6 Conclusions

We identified and formalized the *generative difficulty imbalance* in generative CTR modeling: the uniform treatment of feature fields in the generation objective causes easy fields to dominate training gradients and suppresses learning for informative but hard-to-reconstruct fields. To resolve this, we proposed HeteGenCTR, which introduces a single set of per-field learnable log-difficulty parameters that simultaneously drive two coordinated mechanisms—self-balancing loss aggregation with a provably stable equilibrium, and difficulty-guided attention modulation with scaling coefficients principled to align with the loss-level weighting. The unified signal ensures that both mechanisms remain consistent with the same learned difficulty signal throughout training, creating a holistic solution to the multi-level imbalance. Experiments confirm consistent significant improvements over state-of-the-art baselines.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. 265–283.

[2] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. AutoInt: Automatic Feature Interaction Learning via Self-Attentive Neural Networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM)*. 1161–1170.

[3] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 188–197.

[4] Jianxin Chang, Chenbin Zhang, Yiqun Hui, Dewei Leng, Yanan Niu, Yang Song, and Kun Gai. 2023. PEPNet: Parameter and Embedding Personalized Network for Infusing with Personalized Prior Information. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. 3795–3804.

[5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems (DLRS@RecSys)*. 7–10.

[6] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured Denoising Diffusion Models in Discrete State-Spaces. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*. 17981–17993.

[7] Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. 2021. Conflict-Averse Gradient Descent for Multi-task Learning. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*. 18878–18890.

[8] Kaggle. 2015. Avazu Click-Through Rate Prediction. https://www.kaggle.com/c/avazu-ctr-prediction.

[9] Kaggle. 2014. Criteo Display Advertising Challenge. https://www.kaggle.com/c/criteo-display-ad-challenge.

[10] Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 7482–7491.

[11] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

[12] Jean-Antoine Désidéri. 2012. Multiple-Gradient Descent Algorithm (MGDA) for Multiobjective Optimization. *Comptes Rendus Mathematique*. 350, 5–6 (2012), 313–318.

[13] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*. 794–803.

[14] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine Based Neural Network for CTR Prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*. 2782–2788.

[15] Xingzhuo Guo, Junwei Pan, Ximei Wang, Baixu Chen, Jie Jiang, and Mingsheng Long. 2024. On the Embedding Collapse when Scaling up Recommendation Models. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*.

[16] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*. 249–256.

[17] Tongwen Huang, Zhiqi Zhang, and Junlin Zhang. 2019. FiBiNET: Combining Feature Importance and Bilinear Feature Interaction for Click-Through Rate Prediction. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys)*. 169–177.

[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*. 6840–6851.

[19] Wenjie Wang, Yiyan Xu, Fuli Feng, Xinyu Lin, Xiangnan He, and Tat-Seng Chua. 2023. DiffRec: A Diffusion Collaborative Filtering Framework. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 832–841.

[20] Mengyuan Jing, Yanling Wang, Qing Li, and Chao Wang. 2024. Diffusion Augmentation for Sequential Recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM)*. 912–921.

[21] Fangye Wang, Hansu Gu, Dongsheng Li, Tun Lu, Peng Zhang, and Ning Gu. 2023. Towards Deeper, Lighter and Interpretable Cross Network for CTR Prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM)*. 2523–2533.

[22] Zhiqiang Wang, Qingyun She, and Junlin Zhang. 2021. MaskNet: Introducing Feature-Wise Multiplication to CTR Ranking Models by Instance-Guided Mask. In *Proceedings of DLP-KDD 2021*.

[23] Mingjia Yin, Junwei Pan, Hao Wang, Ximei Wang, Shangyu Zhang, Jie Jiang, Defu Lian, and Enhong Chen. 2025. From Feature Interaction to Feature Generation: A Generative Paradigm of CTR Prediction Models. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*.

[24] Junwei Pan, Wei Xue, Ximei Wang, Haibin Yu, Xun Liu, Shijie Quan, Xueming Qiu, Dapeng Liu, Lei Xiao, and Jie Jiang. 2024. Ads Recommendation in a Collapsed and Entangled World. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. 5566–5577.

[25] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *Proceedings of the 10th International Conference on World Wide Web*. 285–295.

[26] Weinan Zhang, Jiarui Qin, Wei Guo, Ruiming Tang, and Xiuqiang He. 2021. Deep Learning for Click-Through Rate Estimation. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI)*. 4695–4703.

[27] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient Surgery for Multi-Task Learning. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*. 5824–5836.

[28] Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. 2024. Simple and Effective Masked Diffusion Language Models. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems 30 (NIPS)*. 5998–6008.

[30] Hong Wen, Jing Zhang, Fuyu Lv, Wentian Bao, Tianyi Wang, and Zulong Chen. 2021. Hierarchically Modeling Micro and Macro Behaviors via Multi-Task Learning for Conversion Rate Prediction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.

[31] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & Cross Network for Ad Click Predictions. In *Proceedings of the ADKDD Workshop at KDD 2017*. 12:1–12:7.

[32] Ruoxi Wang, Rakesh Shivanna, Derek Zhiyuan Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed H. Chi. 2021. DCN V2: Improved Deep & Cross Network and Practical Lessons for Web-scale Learning to Rank Systems. In *Proceedings of the 30th Web Conference (WWW)*. 1785–1797.

[33] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. 1754–1763.

[34] Steffen Rendle. 2010. Factorization Machines. In *Proceedings of the 10th IEEE International Conference on Data Mining (ICDM)*. 995–1000.
[35] Moyu Zhang, Yujun Jin, Yun Chen, Jinxin Hu, Yu Zhang, and Xiaoyi Zeng. 2026. Infer As You Train: A Symmetric Paradigm of Masked Generative for Click-Through Rate Prediction. In *Proceedings of the ACM Web Conference (WWW)*. 8381-8384.
[36] Moyu Zhang, Yun Chen, Yujun Jin, Jinxin Hu, and Yu Zhang. 2025. DGenCTR: Towards a Universal Generative Paradigm for Click-Through Rate Prediction via Discrete Diffusion. *arXiv preprint arXiv:2508.14500* (2025).
[37] Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Jiayuan He, Yinghai Lu, and Yu Shi. 2024. Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*.