

# Modeling Cascaded Delay Feedback for Online Net Conversion Rate Prediction: Benchmark, Insights and Solutions

Mingxuan Luo\*  
luomingxuan@stu.xmu.edu.cn  
Xiamen University  
Xiamen, China

Guipeng Xv\*  
xuguipeng@stu.xmu.edu.cn  
Xiamen University  
Xiamen, China

Sishuo Chen  
chensishuo.css@alibaba-inc.com  
Taobao & Tmall Group of Alibaba  
Beijing, China

Xinyu Li  
xinyuli@stu.xmu.edu.cn  
Xiamen University  
Xiamen, China

Li Zhang  
zl428934@alibaba-inc.com  
Taobao & Tmall Group of Alibaba  
Beijing, China

Zhangming Chan  
zhangming.czm@alibaba-inc.com  
Taobao & Tmall Group of Alibaba  
Beijing, China

Xiang-Rong Sheng  
xiangrong.sxr@alibaba-inc.com  
Taobao & Tmall Group of Alibaba  
Beijing, China

Han Zhu  
zhuhan.zh@alibaba-inc.com  
Taobao & Tmall Group of Alibaba  
Beijing, China

Jian Xu  
xiyu.xj@alibaba-inc.com  
Taobao & Tmall Group of Alibaba  
Beijing, China

Bo Zheng  
bozheng@alibaba-inc.com  
Taobao & Tmall Group of Alibaba  
Beijing, China

Chen Lin<sup>†</sup>  
chenlin@xmu.edu.cn  
Xiamen University  
Xiamen, China

## Abstract

In industrial recommender systems, conversion rate (CVR) is often used for traffic allocation, but fails to fully reflect recommendation effectiveness as it does not account for refund rate (RFR). Thus, net conversion rate (NetCVR), the probability that a clicked item is purchased and not refunded, is proposed to better show true user satisfaction and business value. Unlike CVR, NetCVR prediction involves a more complex **multi-stage cascaded delay feedback phenomenon**. The two cascaded delays Click → Conversion and Conversion → Refund in NetCVR have opposite effects. Therefore, traditional CVR methods cannot be directly applied. At present, the lack of relevant open-source datasets and online continuous training schemes poses a challenge.

To address these, we first introduce **C**ascadal **S**equences of **C**onversion **A**nd **D**elayed **r**efund (**CASCADE**), the first large-scale open dataset derived from Taobao app for online continuous NetCVR prediction. We further analyze **CASCADE** and derive three key insights: (1) NetCVR exhibits clear temporal patterns necessitating online continuous modeling; (2) Cascaded modeling CVR and RFR for NetCVR outperforms directly modeling NetCVR; and (3) delay time, which correlated with both CVR and RFR, is an important feature for NetCVR prediction. Based on these insights, we propose **neT** convErsion caScaded modeLing and debiAsing method

(**TESLA**). This continuous method features a CVR-RFR cascaded architecture, stage-wise debiasing, and a delay-time-aware ranking loss for efficient NetCVR prediction. Experiments show that **TESLA** outperforms state-of-the-art methods on **CASCADE**, achieving an absolute improvement of **12.41%** in RI-AUC and **14.94%** in RI-PRAUC on NetCVR over the strongest baseline. We hope this work provides a new direction for online delayed feedback modeling in NetCVR prediction. Our code and dataset are available at <https://github.com/alimama-tech/NetCVR>.

## CCS Concepts

• Information systems → Online advertising; • Applied computing → Electronic commerce.

## Keywords

Net Conversion Rate Prediction, Cascaded Delayed Feedback, Continuous Learning

## ACM Reference Format:

Mingxuan Luo, Guipeng Xv, Sishuo Chen, Xinyu Li, Li Zhang, Zhangming Chan, Xiang-Rong Sheng, Han Zhu, Jian Xu, Bo Zheng, and Chen Lin. 2026. Modeling Cascaded Delay Feedback for Online Net Conversion Rate Prediction: Benchmark, Insights and Solutions. In *Proceedings of the ACM Web Conference 2026 (WWW '26)*, April 13–17, 2026, Dubai, United Arab Emirates. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3774904.3792327>

## 1 Introduction

In industrial recommender systems, the post-click conversion rate (CVR) —defined as the probability that a user makes a purchase after clicking an item—is commonly predicted and leveraged for traffic allocation [2, 4, 8, 13, 14, 23–25]. However, CVR alone may not fully capture the true effectiveness of a recommendation algorithm, as

\*The authors contribute equally.

<sup>†</sup> Corresponding author.

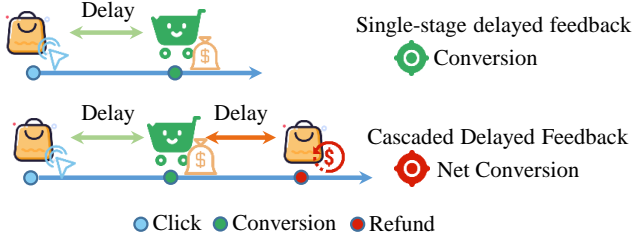


This work is licensed under a Creative Commons Attribution 4.0 International License. *WWW '26, Dubai, United Arab Emirates*

© 2026 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-2307-0/2026/04

<https://doi.org/10.1145/3774904.3792327>



**Figure 1: Illustration of Net Conversion and Conversion. Net conversion shows multi-stage cascaded delay nature, involving Click → Conversion and Conversion → Refund delays.**

users often request refunds when purchased items fail to meet their expectations. To better reflect genuine user satisfaction and business value, it is essential to consider not only whether a purchase occurs, but also whether it is retained —i.e., not followed by a refund. This motivates the prediction of the net conversion rate (NetCVR) [28], which measures the probability that a user completes a purchase and does not subsequently request a refund (as illustrated in Fig. 1).

NetCVR prediction is significantly more challenging than conventional conversion rate (CVR) estimation due to its multi-stage, cascaded delay feedback structure. To obtain a definitive training label, one must observe the full user journey: Click → Conversion → Refund. This process involves two distinct delay intervals: (1) the time between a click and a potential conversion, and (2) the subsequent time between conversion and a potential refund.

Refunds can occur weeks after purchase, but recommender systems need up-to-date models. Existing methods, such as ECAD [28], use fully observed labels and adopt batch-based feedback modeling, typically updating the model once per day offline.

However, offline training fails to capture temporal dynamics and lacks the responsiveness required in real-world recommender systems, where models must continuously adapt to evolving user behavior and market conditions [2, 7, 12, 17, 19, 21, 22, 27]. Meanwhile, offline methods cannot be simply extended to online learning scenarios, as they fail to address issues like stream data modeling and debiasing. Consequently, online continuous approaches to NetCVR prediction remain largely unexplored, despite their critical importance for practical deployment.

In this work, we present **C**ascaded **S**equences of **C**onversion **A**nd **D**elayed **r**efund (**CASCADE**), **the first large-scale open dataset dedicated to online continuous net-conversion prediction**. We derive **CASCADE** using display advertising data collected from the Taobao app, one of the world’s largest e-commerce platforms. It meticulously records the full user interaction sequence, from clicks to conversions and refunds, providing a comprehensive view of user behavior. Each event is timestamped, enabling accurate modeling of label evolution under cascaded delayed feedback. These features make it an invaluable resource for advancing delayed feedback models in NetCVR prediction.

To gather insights for algorithm development, we perform data analysis and experiments on **CASCADE**, leading to three key findings. (1) NetCVR exhibit *distinct temporal patterns*, underscoring the importance of *online continuous training*. (2) Cascaded modeling CVR and refund rate (RFR) for NetCVR outperforms direct

modeling NetCVR, showing significant potential. (3) Delay time correlates with both CVR and RFR. Shorter delays are common in high CVR and high RFR cases, suggesting that delay time could be useful for improving NetCVR prediction.

Based on the above observations, we present **the first on-line continuous method for net conversion prediction, neT conversion caScaded modeLing and debiAsing method (TESLA)**. To address the time-varying nature of NetCVR, **TESLA** employs online continuous training and constructs a novel data stream for NetCVR prediction. Since CVR and RFR are highly correlated, **TESLA** employs a shared-bottom architecture to model CVR and RFR as separate tasks with partial shared parameters. Additionally, a delay-time-aware ranking loss is employed for real-time training on partial feedback. Extension experiments confirm that **TESLA** surpasses state-of-the-art methods on **CASCADE**, achieving an absolute improvement of **12.41%** in RI-AUC and **14.94%** in RI-PRAUC on NetCVR over the strongest baseline.

We hope this work can offer a new direction for online delayed feedback modeling in NetCVR prediction. We summarize our contributions as follows:

- **Data Resources.** We introduce the first public benchmark dataset **CASCADE** for online continuous NetCVR prediction, offering a reproducible streaming experimental environment. We plan to release our code and benchmark post-publication.
- **Valuable Insights.** Based on data and experimental analysis, we reveal the limitations of current methods in handling cascaded delayed feedback and propose several key insights to guide future research directions.
- **Efficient Method.** We evaluate multiple methods on **CASCADE** and introduce the first online continuous training framework specifically designed for NetCVR. Extensive evaluations show that **TESLA** consistently boosts NetCVR prediction performance.

## 2 Related Work

**Conversion Rate Prediction.** Click-through conversion rate (CVR) prediction is a core component in recommender systems and online advertising, directly affecting ranking quality and platform revenue [1, 5, 9, 17, 28]. Existing efforts have advanced CVR modeling from multiple angles, including model architecture design [5, 9, 16], multi-task learning with auxiliary signals [3, 17, 19, 22, 28], and delayed feedback handling [2, 8, 13, 14, 23]. While these methods improve estimation accuracy along the click-to-conversion path, they do not account for post-conversion refund behaviors, a critical factor in determining true user satisfaction and net revenue.

**Delayed Feedback in Conversion Rate Prediction.** Methods for addressing delayed feedback fall into three main paradigms: (1) *Parametric Survival Modeling*: Early approaches such as DFM [2] and NoDeF [26] assume parametric delay distributions (e.g., exponential) and jointly model conversion probability and delay time. (2) *Label Correction with Importance Sampling*: To correct bias from censored negatives, methods like FNW [11] and Defer [8] re-weight unconverted samples upon delayed feedback using importance weighting. (3) *Hybrid Window-Based Methods*: These balance bias and variance by combining unbiased short-term estimation within an observation window and debiased long-term correction beyond it. Examples include Defuse [4] and DDFM [6], while DFSN [14] and

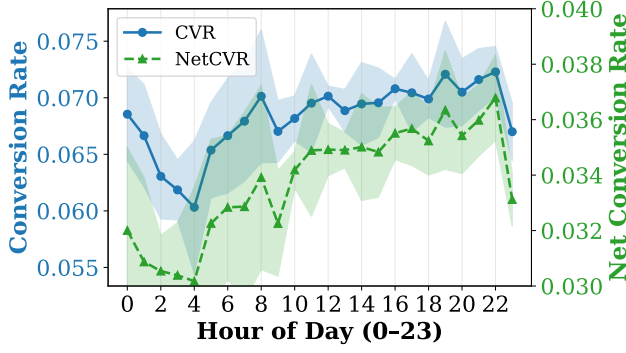


Figure 2: Conversion Rate and Net Conversion Rate by Hours.

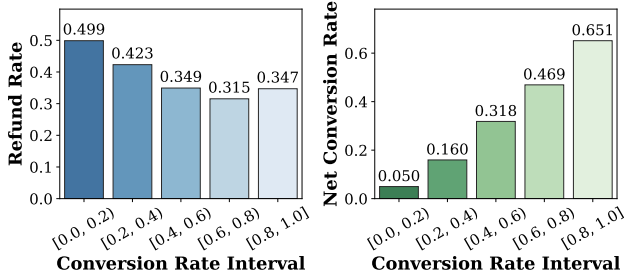


Figure 3: Average Refund Rate and Net Conversion Rate Across Conversion Rate Bins.

MISS [13] further enhance performance via multi-stream modeling across different delay stages.

**Net Conversion and Refund-Aware Modeling.** Recent work has begun to recognize that final user outcomes extend beyond conversion. ECAD [28] introduces net conversion by incorporating refunds, but operates offline with static labels, ignoring the cascaded and asynchronous nature of real-time feedback. In contrast, **TESLA** is the first method designed for online continuous NetCVR prediction under multi-stage cascaded delay feedback.

### 3 Dataset Construction and Analysis

#### 3.1 Dataset Construction

Although the prediction of net conversion rates (NetCVR) is crucial, one limitation is the absence of open-access datasets. Existing datasets such as Criteo<sup>1</sup> and Tencent<sup>2</sup> provide click and conversion data, but none include post-purchase behavioral signals, such as whether a purchase is retained or refunded, making it impossible to accurately model NetCVR. To address these limitations, we have developed the **first open-source dataset for net conversion rate prediction**, named **CASCAD** (Sequences of Conversion And Delayed refund). Detailed construction, feature descriptions, and statistics are provided in Appendix B.

<sup>1</sup><https://labs.criteo.com/2013/12/conversion-logs-dataset/>

<sup>2</sup><https://algo.qq.com/?lang=en>

**Table 1: Net conversion rate performance between the cascade modeling approach and direct modeling approach.**

| Structure | AUC↑   | NLL↓   | PCOC   | PRAUC↑ |
|-----------|--------|--------|--------|--------|
| Cascade   | 0.7525 | 0.1803 | 1.0393 | 0.1585 |
| Direct    | 0.7514 | 0.1814 | 1.2367 | 0.1567 |

#### 3.2 Problem Setup

In the Conversion Rate (CVR), Refund Rate (RFR), and Net Conversion Rate (NetCVR) prediction task, the model processes inputs as  $(\mathbf{x}, y, z) \sim (X, Y, Z)$ . Here,  $\mathbf{x}$  represents features, and  $y \in \{0, 1\}$ ,  $z \in \{0, 1\}$  are the conversion and refund labels. Since net conversion has multi-stage cascaded delay characteristics, we do not collect net conversion labels separately but represent them using conversion and refund labels.  $y = 1$  and  $z = 1$  represent there exist a conversion and a refund, respectively. Net conversion occurs when  $y = 1$  and  $z = 0$ . Given features  $\mathbf{x}$ , the objectives are as follows: For CVR, train a parameterized function  $f_{\theta}^{\text{CVR}}$  to predict the conversion probability; for RFR, train  $f_{\theta}^{\text{RFR}}$  to predict the refund probability; and for NetCVR, train  $f_{\theta}^{\text{NetCVR}}$  to predict the net conversion probability, which can also be represented as  $f_{\theta}^{\text{NetCVR}} = f_{\theta}^{\text{CVR}} * (1 - f_{\theta}^{\text{RFR}})$ . The key notations used in this paper are summarized in Appendix A.

#### 3.3 Insights Exploration

Given the lack of prior art in NetCVR prediction, to gain deeper insights, we employ exploratory data analysis in this section. Key insights are listed as follows:

##### Takeaways

1. NetCVR exhibits clear temporal patterns necessitating online continuous learning.
2. Cascaded modeling CVR and RFR for NetCVR outperforms directly modeling NetCVR.
3. Delay time, which is correlated with CVR and RFR, is an important feature for NetCVR prediction.

**3.3.1 The Significance of Online Continuous Training.** To analyze intra-day patterns of NetCVR targets, we randomly sampled three days' data and computed hourly CVR and NetCVR. CVR was derived by aggregating clicks within each hour and assigning conversion labels based on user behavior over the following 3-day window (conversion attribute window). NetCVR was calculated similarly, with refund labels assigned based on user behavior over a 6-day window (conversion and refund attribute window).

We can observe from Fig. 2 that both CVR and NetCVR display hour-to-hour fluctuations over time. Notably, both indicators experience a substantial decline during late-night periods (e.g., 00:00–04:00). Consequently, relying on offline methods that update models only once per day [28] is insufficient to capture such fine-grained temporal patterns, highlighting the necessity of employing online continuous learning algorithms that can adapt to dynamic user behavior and provide more accurate real-time responses.

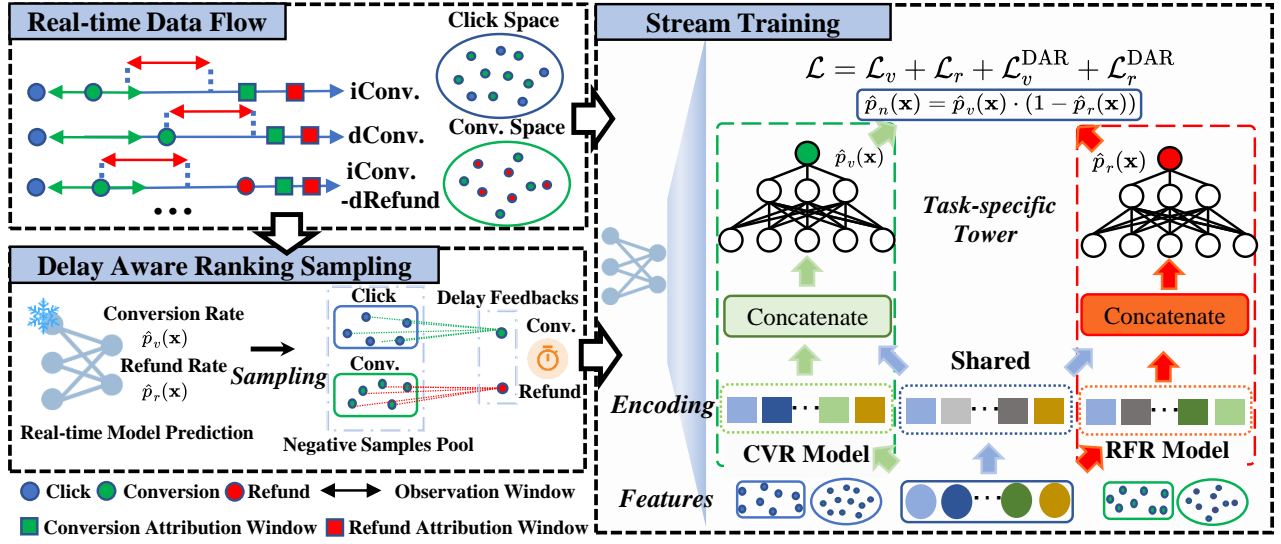


Figure 4: Framework of TESLA. *i* denotes an immediate response while *d* denotes a delayed response.

**3.3.2 The Superiority of Cascade Structure.** We first grouped items by CVR and analyzed their relationships with RFR and NetCVR. As shown in Fig. 3, we can observe that: (1) Items with higher CVR tend to have higher NetCVR, suggesting a possible positive correlation and indicating **potential for joint modeling of CVR and NetCVR with shared parameters**. (2) Items with higher CVR generally show lower RFR. However, RFR increases in the top CVR bucket (0.8, 1], which reveals that **conversion and refund are mostly related but retain some distinct characteristics**.

There are two approaches exist for online continuous NetCVR modeling: (1) direct modeling NetCVR, and (2) cascaded modeling CVR and RFR. Based on above insights, we can use a dual-tower structure to implement both methods. The direct method predicts CVR and NetCVR tasks simultaneously, while the cascaded method predicts CVR and RFR tasks together.

To evaluate these approaches for online NetCVR prediction, we extended the simplest online continual learning method FNC [4, 8] to NetCVR task as there is no prior continuous methods for NetCVR and FNC’s straightforward debiasing strategy can minimize experimental interference. The detail of FNC is shown in AppendixD.4. In this adapted method, conversion events enter the model through the native data flow, and refund events are input when actual refunds occur. We compared AUC, NLL, PRAUC, and PCOC for both variants as proxy metrics to evaluate the performance of netCVR prediction.

As shown in Tab. 1, the cascaded approach attains higher AUC and PR-AUC, signaling superior model ranking accuracy. AUC evaluates the model’s ability to rank true conversions above non-conversions and refunds. (1) Direct modeling NetCVR cannot distinguish between non-conversions and refunds, often overfitting conversion intent features, leading to biased predictions. Thus, its PCOC values exceeding 1. (2) In contrast, a cascaded approach decomposes NetCVR into sequential sub-tasks, i.e., conversion prediction followed by refund prediction, allowing the model to **capture**

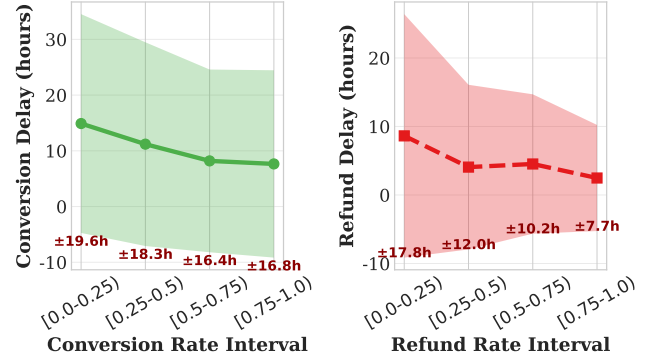


Figure 5: Average conversion and refund delay across CVR and refund rate groups.

**distinct behavioral patterns at each stage and produce more accurate, interpretable estimates.**

**3.3.3 The Potential of Modeling Delay Time for NetCVR Prediction.** We grouped users by CVR and RFR and calculated the mean and variance of delay time for each group. We can observe from Fig. 5 that: (1) High-CVR users have shorter and more consistent payment delays than low-CVR users, suggesting that those who convert quickly may have stronger purchase intent. (2) Compared to the low-RFR group, the high-RFR group has a shorter average delay and smaller variance, which may indicate immediate post-purchase dissatisfaction or that these users are subject to automated return policies. **Delay is not just a technical challenge but a behavioral signal reflecting user confidence and satisfaction.** It should be considered in modeling cascaded delayed feedback.



## 4 Methodology

Based on the gathered insights, we propose **neT conversion caScaded modeLing and debiAsing (TESLA)**. As shown in Fig. 4, **TESLA** consists of three components: (1) Inspired by the insights that NetCVR exhibits clear temporal patterns (§ 3.3.1), **TESLA** adopts an online continuous learning method with two-stage cascaded observation windows to dynamically track net conversion status (§ 4.1); (2) Drawing on the insights that cascaded modeling of CVR and RFR for NetCVR is more effective, and that CVR and RFR are related but have distinct characteristics (§ 3.3.2), **TESLA** employs a partially shared architecture to jointly model the CVR and RFR cascade (§ 4.2). We apply a stage-wise debiasing strategy to keep accuracy (§ 4.3); (3) Further guided by insight that delay time reflects user intent and satisfaction (§ 3.3.3), **TESLA** adopts a **Delay-Aware Ranking loss** to adaptively adjust pairwise loss and emphasize reliable interactions (§ 4.4).

### 4.1 Data Stream

In the complex landscape of NetCVR, traditional CVR prediction data streams fall short due to the multi-stage delayed feedback. To address this, we pioneer a novel stream data structure tailored for net conversion. As illustrated in Fig. 6, **TESLA** establishes a structured pipeline with a dedicated **refund observation window** triggered post-conversion. This secondary window, analogous to the post-click *conversion observation window*, enables early refund signal detection, ensuring accurate and timely NetCVR estimation.

**TESLA** further extends to classifying user interactions into seven trajectory types based on timing patterns. These types can be grouped into two overarching categories: (1) **Conversion with No Refund**, including immediate conversion (**iConv.**), delayed conversion (**dConv.**), and no conversion (**nConv.**). (2) **Conversion with Refund**, including immediate conversion with immediate refund (**iConv.-iRefund**) and with delayed refund (**iConv.-dRefund**); delayed conversion with immediate refund (**dConv.-iRefund**) and with delayed refund (**dConv.-dRefund**).

This refined classification allows the model to effectively differentiate between transient and sustained conversions, which is pivotal for enhancing the accuracy of net conversion estimation and supporting robust debiasing and dynamic label modeling in cascaded delays.

### 4.2 CVR-RFR Cascaded Structure

According to § 3.3.2, direct NetCVR modeling overlooks the dynamic process of Conversion  $\rightarrow$  Refund, leading to poor real-world calibration. Thus, *NetCVR should be modeled via a cascade framework that integrates CVR and RFR*. Given that CVR and RFR are related but have distinct characteristics (§ 3.3.2), the model should capture both their shared and unique features.

Inspired by Progressive Layered Extraction [20], we use a shared-private architecture to jointly learn CVR and RFR while reducing task interference. **TESLA** includes two task-private embedding layers  $\text{Emb}_v$ ,  $\text{Emb}_r$  and one task-shared encoding module  $\text{Enc}_s$ . The private embedding layers model conversion and refund differences while the shared module transfer common knowledge, capturing the relation between the two. After encoding, **TESLA** combines

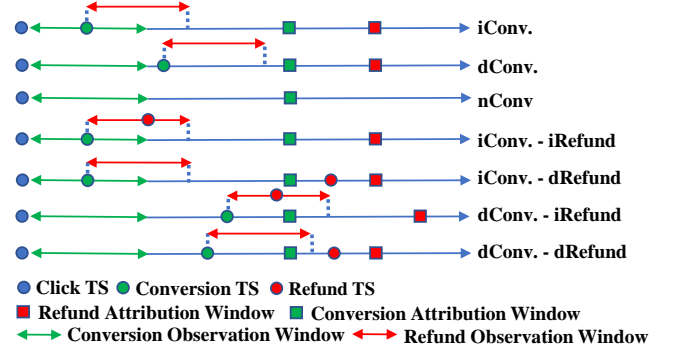


Figure 6: The data flow of **TESLA**. *i* denotes an immediate response while *d* denotes a delayed response.

shared and private information and feeds it into task-specific heads to predict  $\hat{p}_v(\mathbf{x})$  and  $\hat{p}_r(\mathbf{x})$ .

### 4.3 Debiasing Strategies

The cascaded delayed nature of NetCVR introduces system bias into the observed data, affecting accuracy. To achieve an unbiased NetCVR estimate, we use stage-wise importance weighting to debias CVR and RFR: first addressing click-to-conversion delay, then conversion-to-refund delay.

Both the CVR and RFR towers are challenged by delayed labels, necessitating a debiasing approach for accurate estimation. In both towers, a sample is positive only if the event occurs within the observation window; otherwise, it is initially negative. Delayed feedback creates censored positives, which are corrected via a recapture stream upon late observation. However, this delayed correction introduces cascaded delay bias, causing prolonged label uncertainty and temporally misaligned gradients.

To address censored delayed labels in both towers, we employ importance weighting [24] to reweight positive and negative samples. Detailed derivations are provided in the Appendix C. The importance weights for the CVR tower are defined as:

$$w_v^+(\mathbf{x}) = 1 + p_v(\mathbf{x}) \cdot p(h_v > W_v^{\text{obs}} \mid y = 1, \mathbf{x}), \quad (1)$$

$$w_v^-(\mathbf{x}) = \frac{(1 - p_v(\mathbf{x})) \cdot (1 + p_v(\mathbf{x}) \cdot p(h_v > W_v^{\text{obs}} \mid y = 1, \mathbf{x}))}{1 - p_v(\mathbf{x}) + p_v(\mathbf{x}) \cdot p(h_v > W_v^{\text{obs}} \mid y = 1, \mathbf{x})}, \quad (2)$$

where  $w^+$  and  $w^-$  represents the weights for positive and negative samples,  $p_v$  is the conversion probability estimated by the online streaming model.  $p(h_v > W_v^{\text{obs}} \mid y = 1, \mathbf{x})$  is the probability (estimated by a pretrained delay distribution model trained on historical logs follows prior approaches [24]) that a true conversion is delayed beyond the observation window  $W_v^{\text{obs}}$  and thus mislabeled.  $h_v = t_v - t_c$  represents the conversion delay time,  $W_v^{\text{obs}}$  represents the conversion observation window,  $t_v, t_c, t_o$  denotes conversion time, click time and observation time, respectively.  $\mathcal{P}_v$  and  $\mathcal{N}_v$  denote the sets of real-time observed positive and negative conversion samples respectively, which can be estimated via a survival model or empirical statistics. For the RFR tower, analogous importance weights are applied:

$$\begin{aligned} w_r^+(x) &= 1 + p_r(x) \cdot p(h_r > W_r^{obs} \mid z = 1, y = 1, x), \\ w_r^-(x) &= \frac{(1 - p_r(x)) \cdot (1 + p_r(x) \cdot \Pr(h_r > W_r^{obs} \mid z = 1, y = 1, x))}{1 - p_r(x) + p_r(x) \cdot \Pr(h_r > W_r^{obs} \mid z = 1, y = 1, x)}, \end{aligned} \quad (3)$$

where  $p_r$  denotes the refund probability estimated by the online streaming model.  $p(h_r > W_r^{obs} \mid z = 1, y = 1, x)$  represents the probability that a true refund ( $z = 1$ ) is delayed beyond the observation window ( $h_r > W_r^{obs}$ ). It is also estimated by a delay distribution model trained on historical logs.  $h_r = t_r - t_v$  represents the refund delay time,  $W_r^{obs}$  represents the refund observation window,  $t_r$  denotes refund time.  $\mathcal{P}_r$  and  $\mathcal{N}_r$  denote the observed positive and negative refund samples among converted instances.

The debiased loss functions for both towers are formulated as follows. For the CVR tower:

$$\mathcal{L}_v = - \sum_{(x,y) \in \mathcal{P}_v \cup \mathcal{N}_v} [y \cdot w_v^+(x) \cdot \log \hat{p}_v(x) + (1-y) \cdot w_v^-(x) \cdot \log (1 - \hat{p}_v(x))], \quad (5)$$

For the RFR tower,

$$\mathcal{L}_r = - \sum_{(x,z) \in \mathcal{P}_r \cup \mathcal{N}_r} [z \cdot w_r^+(x) \cdot \log \hat{p}_r(x) + (1-z) \cdot w_r^-(x) \cdot \log (1 - \hat{p}_r(x))], \quad (6)$$

This two-stage debiasing framework ensures asymptotically unbiased NetCVR estimation by enabling  $\hat{p}_v(x)$  and  $\hat{p}_r(x)$  to approach their respective ground-truth expectations  $\hat{p}_n(x) = \hat{p}_v(x) \cdot (1 - \hat{p}_r(x))$ . By stage-based bias correction, **TESLA** effectively manages the dual-delay structure in cascaded feedback, enhancing the applicability of single-stage debiasing techniques to the intricate NetCVR setting.

#### 4.4 Delay-Aware Ranking Loss

Online CVR and NetCVR models are typically trained by point-wise loss. However, delayed feedback, especially the multi-stage cascaded delays in NetCVR prediction, can cause label uncertainty and potential label flipping. Point-wise loss struggles to capture this uncertainty, limiting its effectiveness in real-world scenarios.

To address this, we introduce pair-wise loss, which emphasizes ranking positive samples higher than negative ones. However, the inconsistent quality of samples can undermine its effectiveness. Fortunately, our analysis in Section 3.3.3 reveals a relationship between delay time and conversion/refund behavior, which provides a solution. By leveraging delay time as a signal, we can increase the weight of high-quality positive samples and select more reliable negative samples.

**4.4.1 Delay-Based Positive Sample Weighting.** We employ a reweighting approach to differentially treat positive samples. For a batch of observed positive conversion samples  $\mathcal{P}_v$ , let  $\mathbf{h}_v = \{h_v^{(i)}\}_{i \in \mathcal{P}_v}$  denote their corresponding conversion delays, where  $h_v^{(i)} = t_v^{(i)} - t_c^{(i)}$  is the delay time of each sample,  $t_v, t_c$  are the conversion time and click time. Since *high-CVR items convert faster with smaller delay variance* (§ 3.3.3), we assign each positive sample a weight based on its delay:

$$w_i = w(h_v^{(i)}) = w_{\min} + \alpha \cdot \sigma((m - h_v^{(i)})/s), \quad (7)$$

where  $m = \text{median}(\mathbf{h}_v)$  is the median conversion latency in the batch,  $s = \max(\text{std}(\mathbf{h}_v), 1)$  measures delay dispersion. We clamp

$s$  to prevent overconfidence due to small variance.  $\sigma(\cdot)$  is the sigmoid function and  $w_{\min} > 0$  to ensure slow converters still contribute learning signals.  $\alpha$  is a hyper-parameter. This design allocates greater weights to fast conversions samples.

**4.4.2 Uncertainty-Based Negative Sample Sampling.** We replace the random negative sampling process in ranking tasks. Here, we assume that the lower the predicted CVR, the higher the likelihood of a sample being a true negative. we define the sampling probability and normalize it as follows:

$$\pi_j = \frac{(1 - \hat{p}_v(x_j))/\tau}{\sum_{k \in \mathcal{N}_v} [(1 - \hat{p}_v(x_k))/\tau]}, \quad j \in \mathcal{N}_v, \quad (8)$$

where  $\tau > 0$  is a temperature hyperparameter that controls the softness of the sampling process.  $\hat{p}_v$  is the conversion probability,  $\pi$  is the sample probability. We then independently sample  $K$  negative items for each positive item  $i \in \mathcal{P}_v$  according to the distribution  $\pi$ , forming  $K$  pairs  $\{(i, j_1), \dots, (i, j_K)\}$ .

**4.4.3 Delay-aware Ranking Loss.** The loss for the conversion stage is calculated as:

$$\mathcal{L}_v^{\text{DAR}} = \sum_{i \in \mathcal{P}_v} w_i \cdot \left( -\frac{1}{K} \sum_{j \in \mathcal{N}_v, k=1}^K \log \sigma(o_i^{\text{CVR}} - o_{j_k}^{\text{CVR}}) \right), \quad (9)$$

where  $w_i$  is the weight for positive sample  $i$ ,  $o_i^{\text{CVR}} = f_{\theta}^{\text{CVR}}(\mathbf{x}_i)$  denote the output for positive sample  $i$ , and  $\{o_j\}_{k=1}^K$  denote the logit outputs of its sampled negatives. This loss function combines delay-based positive weighting and uncertainty-aware negative sampling, enabling more accurate and stable preference learning under conversion delay. Similarly, we apply the same approach to refunds,

$$\mathcal{L}_r^{\text{DAR}} = \sum_{i \in \mathcal{P}_r} w_i \cdot \left( -\frac{1}{K} \sum_{j \in \mathcal{N}_r, k=1}^K \log \sigma(o_i^{\text{RFR}} - o_{j_k}^{\text{RFR}}) \right), \quad (10)$$

where key conversion-related variables are substituted with their refund counterparts,  $\mathcal{P}_v, \mathcal{N}_r$  are the positive and negative refund sample set,  $o_i^{\text{RFR}} = f_{\theta}^{\text{RFR}}(\mathbf{x}_i)$  denote the output for sample  $i$ . This highlights the symmetric and versatile nature of Delay-aware Ranking Loss in handling both stages of cascaded delayed feedback.

The final loss consists of the debiased CVR loss, debiased RFR loss, delay-aware CVR ranking Loss and delay-aware RFR ranking Loss,

$$\mathcal{L} = \mathcal{L}_v + \mathcal{L}_r + \mathcal{L}_v^{\text{DAR}} + \mathcal{L}_r^{\text{DAR}}, \quad (11)$$

## 5 Experiments

In this section, we study the following research questions: **RQ1:** Can **TESLA** improve the performance of net conversion rate prediction? (§ 5.2) **RQ2:** How does each component of **TESLA** affect the performance? Specifically, we perform ablation study on "Cascaded structure" (§ 5.3.1), "Debiasing strategies" (§ 5.3.2), and "Delay-aware ranking loss" (§ 5.3.3). **RQ3:** How does observation window settings affect the search performance? (§ 5.4)

### 5.1 Experimental Setting

**5.1.1 Data Settings & Sample Delivery.** To simulate streaming training, we follow the online-learning paradigm [4, 8], split the data into pre-train and streaming chunks, and design a four-window mechanism together with a corresponding sample-delivery rule. Further details are given in Appendix D.1 and Appendix D.2.

**Table 2: Performance Comparison on CVR and NetCVR prediction. Oracle is the upper bound on the best achievable performance as there are no delayed Conversions. "Shared", "Separate", and "Hybrid" are three variants of TESLA. Bold indicates the best results and the oracle results for reference are grayed out.**

| Model                   | AUC↑<br>(CVR / NetCVR) | NLL↓<br>(CVR / NetCVR) | PRAUC↑<br>(CVR / NetCVR) | RI-AUC↑<br>(CVR / NetCVR) | RI-PRAUC↑<br>(CVR / NetCVR) |
|-------------------------|------------------------|------------------------|--------------------------|---------------------------|-----------------------------|
| <b>Pre-trained</b>      | 0.7223 / 0.7353        | 0.2252 / 0.1810        | 0.1768 / 0.1521          | ~                         | ~                           |
| <b>Oracle</b>           | 0.7556 / 0.7643        | 0.2155 / 0.1750        | 0.2007 / 0.1722          | ~                         | ~                           |
| <b>BDL</b>              | 0.7381 / 0.7476        | 0.2237 / 0.1805        | 0.1832 / 0.1574          | 47.45% / 42.41%           | 26.78% / 26.37%             |
| <b>FNC</b>              | 0.7437 / 0.7525        | 0.2225 / 0.1801        | 0.1852 / 0.1585          | 64.26% / 59.31%           | 35.15% / 31.84%             |
| <b>FNW (RecSys'19)</b>  | 0.7443 / 0.7539        | 0.2220 / 0.1800        | 0.1887 / 0.1619          | 66.07% / 64.14%           | 49.79% / 48.76%             |
| <b>DEFER (KDD'21)</b>   | 0.7450 / 0.7552        | 0.2190 / 0.1777        | 0.1920 / 0.1649          | 68.17% / 68.62%           | 63.60% / 63.68%             |
| <b>DEFUSE (WWW'22)</b>  | 0.7459 / 0.7550        | 0.2347 / 0.1971        | 0.1925 / 0.1645          | 70.87% / 67.93%           | 65.69% / 61.69%             |
| <b>ES-DFM (AAAI'21)</b> | 0.7456 / 0.7546        | 0.2190 / 0.1783        | 0.1924 / 0.1644          | 69.97% / 66.55%           | 65.27% / 61.19%             |
| <b>DSFN (SIGIR'23)</b>  | 0.7437 / 0.7527        | 0.2199 / 0.1793        | 0.1905 / 0.1617          | 64.26% / 60.00%           | 57.32% / 47.76%             |
| <b>DDFM (CIKM'23)</b>   | 0.7446 / 0.7541        | 0.2199 / 0.1793        | 0.1918 / 0.1640          | 66.97% / 64.83%           | 62.76% / 59.20%             |
| <b>MISS (AAAI'24)</b>   | 0.7441 / 0.7527        | 0.2199 / 0.1783        | 0.1902 / 0.1625          | 65.47% / 60.00%           | 56.07% / 51.74%             |
| <b>TESLA (Separate)</b> | 0.7456 / 0.7555        | 0.2190 / 0.1776        | 0.1924 / 0.1651          | 69.97% / 69.66%           | 65.27% / 64.68%             |
| <b>TESLA (Shared)</b>   | 0.7458 / 0.7567        | 0.2189 / 0.1772        | 0.1926 / 0.1667          | 70.57% / 73.79%           | 66.11% / 72.64%             |
| <b>TESLA (Hybrid)</b>   | 0.7469 / 0.7579        | 0.2188 / 0.1771        | 0.1933 / 0.1676          | 73.87% / 77.93%           | 69.04% / 77.11%             |
| <b>TESLA</b>            | <b>0.7479 / 0.7588</b> | <b>0.2182 / 0.1767</b> | <b>0.1939 / 0.1679</b>   | <b>76.88% / 81.03%</b>    | <b>71.55% / 78.61%</b>      |

**5.1.2 Evaluation Metrics.** We use AUC, PRAUC, NLL, RI-AUC, and RI-PRAUC to evaluate NetCVR prediction performance [4, 8, 13]. The detailed metrics are shown in Appendix D.3.

**5.1.3 Competitors.** We compare against three reference variants: **Pre-trained**, a static model without online updates; **Oracle**, a privileged model trained with complete feedback after full observation; and **BDL**, an offline model updated daily using complete labels after the attribution window. In addition, we benchmark **TESLA** against eight strong delayed-feedback baselines, including label-correction methods (FNC, FNW, ES-DFM, DEFER, DEFUSE) and hybrid multi-window approaches (DDFM, DSFN, MISS). Detailed descriptions are provided in Appendix D.4.

## 5.2 Performance Comparison

We evaluate all methods on the **CASCADE** dataset. We have following observations from Tab. 2: (1) **TESLA** consistently enhances both CVR and NetCVR tasks. It outperforms other methods across all metrics, showing its effectiveness in tackling delayed feedback and better capturing user preferences. The improvements of RI-AUC in the NetCVR task (12.41%) are more pronounced than those in the CVR task (6.01%), suggesting that it is **more effective in addressing the multi-stage cascaded delayed feedback**. (2) **Continuous training contributes to the performance lift**. All online models perform better than BDL models, which validates the need for online continuous learning (§ 3.3.1) and demonstrates the advantage of online continuous learning models in promptly updating user preferences for better recommendations. (3) Adding Delay-aware Ranking Loss proves effective. Compared with **TESLA** (Hybrid), **TESLA** improves RI-AUC by 3.01% in CVR and 3.1% in NetCVR. This shows that **using delay time helps online models better identify delayed labels**.

## 5.3 Ablation Study

**5.3.1 Impact of Cascaded Structure.** To explore the impact of cascaded structure, we have three variants: (1) Fully shared model (**Shared**): CVR and RFR towers share all lower-layer parameters. (2) Fully separate model (**Separate**): CVR and RFR towers have separate lower-layer parameters. (3) Hybrid shared-independent model (Our method, **Hybrid**): CVR and RFR towers combine shared and task-specific lower-layer parameters. We remove the delay-aware ranking loss to eliminate potential interference.

From Tab. 2 we can observe that: (1) **Hybrid** achieves the best results. By separating shared and unique parameters, **Hybrid** effectively balances knowledge transfer and task specificity. This validates the findings in § 3.3.2, which highlight that CVR and RFR share relevant information yet retain unique content, making this approach crucial for accurate NetCVR estimation. (2) **Shared** performs better than **Separate**. Given the limited availability of refund data, parameter sharing enhances the performance of the RFR model, surpassing the results of completely independent modeling.

**5.3.2 Impact of Refund De-biasing.** To explore the impact of refund debiasing, we conduct ablation studies by removing the Refund Observation Window (ROW) and debiasing strategy (DS) under **Separate**. Note that DS can only be used with ROW enabled.

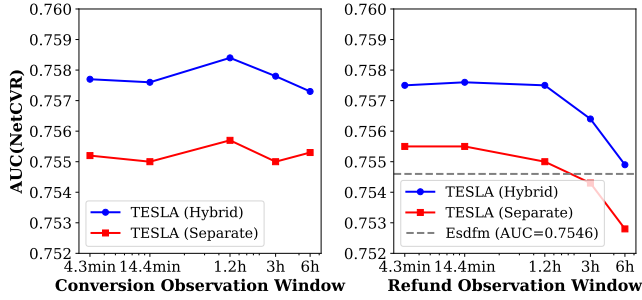
We can observe from Tab. 3 that: (1) Adding ROW alone enhances AUC and reduces NLL, demonstrating improved ranking accuracy and lower uncertainty. This indicates that **the observation window effectively filters early refunds**, thereby reducing label noise. (2) Combining both ROW and DS achieves optimal performance. This suggests that while early filtering with ROW is beneficial, **an explicit debiasing strategy is indispensable** for precise NetCVR estimation.

**Table 3: Ablation study on the Refund Observation Window (ROW) and Refund Debiasing Strategy (DS), built upon the Separate architecture in § 5.3.1.**

| ROW | DS | AUC $\uparrow$ | NLL $\downarrow$ | PRAUC $\uparrow$ |
|-----|----|----------------|------------------|------------------|
| ×   | ×  | 0.7546         | 0.1783           | 0.1644           |
| ✓   | ×  | 0.7553         | 0.1779           | 0.1644           |
| ✓   | ✓  | 0.7555         | 0.1776           | 0.1651           |

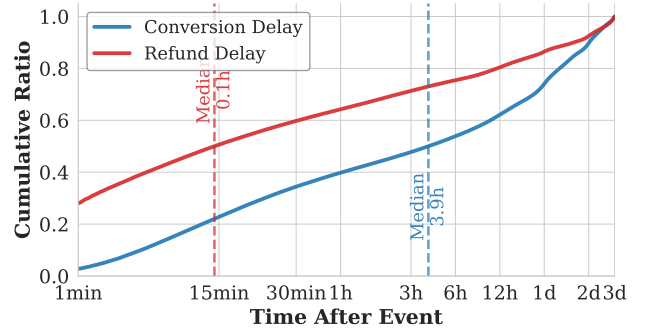
**Table 4: Ablation study on Delay-Aware Ranking Loss (DAR), built upon the Hybrid architecture in § 5.3.1. NR represents no ranking loss. PW, RN, LN, and HN denote positive weighting, random, low-uncertainty, and high-uncertainty negative sampling, respectively.**

| Variation   | RI-AUC $\uparrow$<br>(CVR / NetCVR) | NLL $\downarrow$<br>(CVR / NetCVR) | RI-PRAUC $\uparrow$<br>(CVR / NetCVR) |
|-------------|-------------------------------------|------------------------------------|---------------------------------------|
| NR          | 73.87% / 77.93%                     | 0.2188 / 0.1771                    | 69.04% / 77.11%                       |
| RN          | 75.68% / 79.77%                     | 0.2186 / 0.1767                    | 69.60% / 77.28%                       |
| HN          | 75.08% / 76.44%                     | 0.2226 / 0.1801                    | 69.60% / 76.29%                       |
| LN          | 76.08% / 80.23%                     | 0.2182 / 0.1767                    | 70.85% / 78.23%                       |
| DAR (PW-LN) | <b>76.78% / 80.92%</b>              | <b>0.2182 / 0.1767</b>             | <b>71.41% / 78.77%</b>                |

**Figure 7: Ablation study on refund and conversion observation window.**

**5.3.3 Impact of Delay-aware Ranking Loss.** To explore the effectiveness of delay-aware ranking loss (DAR), we assess its key components—delay-aware positive weighting (PW) and low-uncertainty-driven negative sampling (LN) within the **Hybrid** in § 5.3.1. The variants include: **NR** (no ranking loss), **RN** (random negatives with ranking loss), **HN** (high-uncertainty negative sampling), **LN** (low-uncertainty negative sampling), and **DAR (PW-LN)** (our method with positive weighting and low-uncertainty negatives).

From Tab. 4, we observe that: (1) **DAR** performs best, as it reduces potential noise in both positive and negative samples. (2) Randomly negatives (**RN**) outperforms high-uncertainty negatives (**HN**) but underperforms low-uncertainty negatives (**LN**), underscoring the importance of uncertainty-guided negative sampling. (3) **PW-LN** is better than **LN**, demonstrating the effectiveness of positive sample weighting and the value of focusing on small delay times to capture high-confidence user intent (§ 3.3.3). (4) **NR** performs the worst, indicating that pair-wise ranking loss can mitigate label uncertainty caused by delayed feedback (§ 4.4).

**Figure 8: Cumulative distribution of conversion and refund delays.**

## 5.4 Impact of Observation Windows

To optimize NetCVR performance through selecting and tuning the conversion and refund observation windows, we conduct parameter experiments on  $W_v^{obs}$  and  $W_r^{obs}$  over  $\{0.003, 0.01, 0.05, 0.125, 0.25\}$  days, modifying one window at a time while fixing the other at 0.01 days. AUC (NetCVR) serves as the primary evaluation metric.

From Fig. 7, we can observe that: (1) Conversion observation window  $W_v^{obs}$  are achieved at 0.05 days, with performance declining beyond this point. This highlights the need for a balance between sufficient time to gather stable positive signals and maintaining freshness, reflecting the slower and more diverse conversion dynamics as shown in Fig. 8. (2) In contrast, shorter refund observation windows  $W_r^{obs}$  are preferable. AUC remains stable when  $W_r^{obs} \leq 0.01$  days and declines with longer durations. When the window is too long, the results may even be worse than those without an refund observation window. This aligns with the refund delay distribution in Fig. 8, where most refunds occur quickly. **Shorter window enhances label purity by excluding delayed cases, improving training reliability.**

## 6 Conclusion

In this paper, We study multi-stage cascaded delayed feedback in NetCVR prediction and introduce **CASCADE**, the **first large-scale open dataset for continuous NetCVR modeling** from Taobao. Analysis reveals key temporal patterns and the benefit of cascaded modeling. Based on these insights, we propose **TESLA**, the **first online continuous learning method addressing cascaded delayed feedback** with a CVR-RFR structure, stage-aware debiasing, and delay-aware ranking loss. Experiments show its superior performance over existing methods. We hope that our exploration will be helpful for future research in this area.

## Acknowledgments

This work was supported by the Natural Science Foundation of China (No.62432011,62372390), Science Fund for Distinguished Young Scholars of Fujian Province (No.2025J010001), and Alibaba Group through Alibaba Innovative Research Program. Chen Lin is the corresponding author. This work was completed while the first author was an intern at Alibaba.



## References

- [1] Zhangming Chan, Yu Zhang, Shuguang Han, Yong Bai, Xiang-Rong Sheng, Siyuan Lou, et al. 2023. Capturing conversion rate fluctuation during sales promotions: A novel historical data reuse approach. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3774–3784.
- [2] Olivier Chapelle. 2014. Modeling delayed feedback in display advertising. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1097–1105.
- [3] Sishuo Chen, Zhangming Chan, Xiang-Rong Sheng, Lei Zhang, Sheng Chen, et al. 2025. See Beyond a Single View: Multi-Attribution Learning Leads to Better Conversion Rate Prediction. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management* (Seoul, Republic of Korea) (CIKM '25). Association for Computing Machinery, New York, NY, USA, 5600–5608.
- [4] Yu Chen, Jiaqi Jin, Hui Zhao, Pengjie Wang, Guojun Liu, Jian Xu, and Bo Zheng. 2022. Asymptotically unbiased estimation for delayed feedback modeling via label correction. In *Proceedings of the ACM Web Conference 2022*. 369–379.
- [5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishu Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ipsir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.
- [6] Sunhao Dai, Yuqi Zhou, Jun Xu, and Ji-Rong Wen. 2023. Dually enhanced delayed feedback modeling for streaming conversion rate prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 390–399.
- [7] Ke Fei, Xinyue Zhang, and Jingjing Li. 2025. Entire-Space Variational Information Exploitation for Post-Click Conversion Rate Prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 11654–11662.
- [8] Siyu Gu, Xiang-Rong Sheng, Ying Fan, Guorui Zhou, and Xiaoqiang Zhu. 2021. Real negatives matter: Continuous training with real negatives for delayed feedback modeling. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2890–2898.
- [9] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).
- [10] Sergey Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *JMLR.org* (2015).
- [11] Sofia Ira Ktena, Alykhan Tejani, Lucas Theis, Pranay Kumar Myana, Deepak Dilipkumar, Ferenc Huszár, Steven Yoo, and Wenzhe Shi. 2019. Addressing delayed feedback for continuous training with neural networks in CTR prediction. In *Proceedings of the 13th ACM conference on recommender systems*. 187–195.
- [12] Kuang-chih Lee, Burkay Orten, Ali Dasdan, and Wentong Li. 2012. Estimating conversion rate in display advertising from past performance data. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 768–776.
- [13] Qiming Liu, Xiang Ao, Yuyao Guo, and Qing He. 2024. Online conversion rate prediction via multi-interval screening and synthesizing under delayed feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 8796–8804.
- [14] Qiming Liu, Haoming Li, Xiang Ao, Yuyao Guo, Zhihong Dong, Ruobing Zhang, Qiong Chen, Jianfeng Tong, and Qing He. 2023. Online Conversion Rate Prediction via Neural Satellite Networks in Delayed Feedback Advertising. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1406–1415.
- [15] Ilya Loshchilov and Frank Hutter. 2017. Decoupled Weight Decay Regularization. (2017).
- [16] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1930–1939.
- [17] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. 2018. Entire space multi-task model: An effective approach for estimating post-click conversion rate. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 1137–1140.
- [18] Maas and Hannun. [n. d.]. Rectifier nonlinearities improve neural network acoustic models. ([n. d.]).
- [19] Hongzu Su, Lichao Meng, Lei Zhu, Ke Lu, and Jingjing Li. 2024. DDPO: Direct dual propensity optimization for post-click conversion rate estimation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1179–1188.
- [20] Hongyan Tang, Junling Liu, Ming Zhao, and Xudong Gong. 2020. Progressive Layered Extraction (PLE): A Novel Multi-Task Learning (MTL) Model for Personalized Recommendations. In *RecSys '20: Fourteenth ACM Conference on Recommender Systems*.
- [21] Guoquan Wang, Qiang Luo, Weisong Hu, Pengfei Yao, Wencong Zeng, Guorui Zhou, and Kun Gai. 2025. FIM: Frequency-Aware Multi-View Interest Modeling for Local-Life Service Recommendation. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1748–1757.
- [22] Hao Wang, Tai-Wei Chang, Tianqiao Liu, Jianmin Huang, Zhichao Chen, Chao Yu, Ruopeng Li, and Wei Chu. 2022. ESCM2: Entire space counterfactual multi-task model for post-click conversion rate estimation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 363–372.
- [23] Yifan Wang, Peijie Sun, Min Zhang, Qinglin Jia, Jingjie Li, and Shaoping Ma. 2023. Unbiased delayed feedback label correction for conversion rate prediction. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2456–2466.
- [24] Jia-Qi Yang, Xiang Li, Shuguang Han, Tao Zhuang, De-Chuan Zhan, Xiaoyi Zeng, and Bin Tong. 2021. Capturing delayed feedback in conversion rate prediction via elapsed-time sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4582–4589.
- [25] Shota Yasui, Gota Morishita, Fujita Komei, and Masashi Shibata. 2020. A feedback shift correction in predicting conversion rates under delayed feedback. In *Proceedings of The Web Conference 2020*. 2740–2746.
- [26] Yuya Yoshikawa and Yusaku Imai. 2018. A Nonparametric Delayed Feedback Model for Conversion Rate Prediction. CoRR abs/1802.00255 (2018). *arXiv preprint arXiv:1802.00255* (2018).
- [27] Xinyue Zhang, Cong Huang, Kun Zheng, Hongzu Su, Tianxu Ji, Wei Wang, Hongkai Qi, and Jingjing Li. 2024. Adversarial-Enhanced Causal Multi-Task Framework for Debiasing Post-Click Conversion Rate Estimation. In *Proceedings of the ACM Web Conference 2024*. 3287–3296.
- [28] Yunfeng Zhao, Xu Yan, Xiaoqiang Gui, Shuguang Han, Xiang-Rong Sheng, et al. 2023. Entire space cascade delayed feedback modeling for effective conversion rate prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 4981–4987.

## A Key Notations

The key notations used throughout this paper are summarized in Table 5.

Table 5: Notation and explanations

| Notation  | Explanation  |
|---|--|
| $\mathbf{x}, \mathbf{y}, \mathbf{z}$                                  | Input features, conversion labels, refund labels   |
| $t_c, t_v, t_r, t_o$  | click time, conversion time, refund time, observation time   |
| $e_v, e_r$  | $e_v = t_o - t_c$ : interval between click and observation time; $e_r = t_o - t_v$ : interval between conversion and observation time  |
| $h_v, h_r$  | conversion delay: $h_v = t_v - t_c$ ; refund delay: $h_r = t_r - t_o$  |
| $W_v^{obs}, W_r^{obs}$<br>$W_v^{attr}, W_r^{attr}$                    | conversion/refund observation window<br>conversion/refund Attribution window   |
| $p_v(\mathbf{x}), q_v(\mathbf{x})$                                    | ground-truth of conversion: $p_v(\mathbf{x}) = p(y = 1 \mathbf{x})$ , observed distribution of conversion: $q_v(\mathbf{x}) = q(y = 1 \mathbf{x})$   |
| $p_r(\mathbf{x}), q_r(\mathbf{x})$                                    | ground-truth of refund: $p_r(\mathbf{x}) = p(z = 1 \mathbf{y} = 1, \mathbf{x})$ , observed distribution of refund: $q_r(\mathbf{x}) = q(z = 1 \mathbf{y} = 1, \mathbf{x})$   |
| $p_n(\mathbf{x}), q_n(\mathbf{x})$                                    | ground-truth of net conversion: $p_n(\mathbf{x}) = p_v(\mathbf{x}) \cdot (1 - p_r(\mathbf{x}))$ , observed distribution of net conversion: $q_n(\mathbf{x}) = q_c(\mathbf{x}) \cdot (1 - q_r(\mathbf{x}))$   |
| $\mathcal{P}_v, \mathcal{P}_r$  | real-time positive conversion/refund samples set from streaming data   |
| $\mathcal{N}_v, \mathcal{N}_r$  | real-time negative conversion/refund samples set from streaming data   |
| $\hat{p}_v(\mathbf{x}), \hat{p}_r(\mathbf{x}), \hat{p}_n(\mathbf{x})$ | output logits of the CVR and RFR models, i.e., $\hat{p}_v(\mathbf{x}) = f_{\theta}^{\text{CVR}}(\mathbf{x})$ , $\hat{p}_r(\mathbf{x}) = f_{\theta}^{\text{RFR}}(\mathbf{x})$ , $\hat{p}_n(\mathbf{x}) = \hat{p}_v(\mathbf{x}) \cdot (1 - \hat{p}_r(\mathbf{x}))$ |

## B Dataset Construction

**CASCADE** not only includes clicks and conversions but also incorporates refund information, along with timestamps for all interactions. These features allow for accurate modeling of NetCVR prediction with multi-stage cascaded delayed feedback and also provide benefits for CVR and RFR prediction tasks. The dataset covers sampled clicks in 25 days, and the max attribution window is 15 days. The statistics are summarized in Tab. 6.

### B.1 Data Collection

**CASCADE** is collected from Taobao, one of the largest E-commerce sites worldwide, designed to support research on post-click value prediction under realistic online continuous learning conditions. To ensure behavioral diversity and representativeness, we sample from multiple advertising scenarios across different traffic and intent levels, covering both high-exposure placements and intent-rich contexts. Users are stratified by activity level and sampled to maintain balanced behavioral coverage over time, forming a stable, million-scale training set. To reflect the characteristics of NetCVR prediction, we upsample active users exhibiting high refund rates. This dataset does not represent real-world business metrics or operational conditions.

### B.2 Feature Profile

Each ad click instance includes two types of signals: (1) *Categorical features*, consisting of 22 fields in total: 8 user features (e.g., age groups and membership levels), 6 item features (e.g., category, brand), and 8 context features. All identifiers are irreversibly hashed to ensure privacy. (2) *Temporal information*, including the click timestamp and the sequence of subsequent conversion and refund timestamps, which enable modeling of delayed feedback and dynamic user behavior.

### B.3 Data Compliance and Availability

All personally identifiable and business-sensitive information, such as raw user, item, campaign, and placement IDs, is rigorously de-identified or removed. Only features essential for modeling are retained, and the dataset does not reflect real-time business operations. We will release **CASCADE** as a public benchmark to facilitate research on netCVR prediction with privacy-preserving, real-world data.

## C Debiasing Strategy in ES-DFM

In streaming conversion rate prediction, due to delayed feedback, the observed label distribution  $q(y|\mathbf{x})$  is biased. Positive conversions that occur after the observation time are mislabeled as negatives, leading to underestimation of the true conversion rate  $p_v(\mathbf{x}) = p(y = 1|\mathbf{x})$ .

To achieve unbiased estimation, ES-DFM adopt importance sampling under the *elapsed-time sampling* framework. The key idea is to model the elapsed time  $e_v = t_o - t_v$  as a random variable with distribution  $p(e_v|\mathbf{x})$ , and then correct for selection bias via reweighting.

Due to delayed feedback, only conversions with  $h_c \leq e_v$  are observed. Hence, the observed distribution is:

$$q(y = 1|\mathbf{x}) = p(y = 1|\mathbf{x}) \cdot p(h_v \leq e_v | y = 1, \mathbf{x}), \quad (12)$$

$$q(y = 0|\mathbf{x}) = p(y = 0|\mathbf{x}) + p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x}). \quad (13)$$

In ES-DFM, when a delayed conversion is later observed, It insert a positive duplicate into the training stream. This increases the total number of samples by  $p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x})$ . Therefore, the observed distribution must be normalized by  $1 + p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x})$ , yielding:

$$q(y = 1|\mathbf{x}) = \frac{p(y = 1|\mathbf{x})}{1 + p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x})}, \quad (14)$$

$$q(y = 0|\mathbf{x}) = \frac{p(y = 0|\mathbf{x}) + p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x})}{1 + p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x})}, \quad (15)$$

ES-DFM apply importance sampling to recover the expectation under the true distribution  $p(y|\mathbf{x})$ :

$$\begin{aligned} \mathcal{L}_{\text{ideal}} &= \mathbb{E}_{(\mathbf{x}, y) \sim p(\mathbf{x}, y)} [\ell(y, \hat{p}_v(\mathbf{x}))], \\ &= \mathbb{E}_{(\mathbf{x}, y) \sim q(\mathbf{x}, y)} \left[ \frac{p(y|\mathbf{x})}{q(y|\mathbf{x})} \cdot \ell(y, \hat{p}_v(\mathbf{x})) \right], \end{aligned} \quad (16)$$

The importance weights are computed as follows:

For positive samples ( $y = 1$ ):

$$\begin{aligned} w_v^+ &= \frac{p(y = 1|\mathbf{x})}{q(y = 1|\mathbf{x})}, \\ &= \frac{p(y = 1|\mathbf{x}) \cdot (1 + p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x}))}{p(y = 1|\mathbf{x})}, \\ &= 1 + p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x}), \end{aligned} \quad (17)$$

For negative samples ( $y = 0$ ):

$$\begin{aligned} w_v^- &= \frac{p(y = 0|\mathbf{x})}{q(y = 0|\mathbf{x})}, \\ &= \frac{p(y = 0|\mathbf{x})}{\frac{p(y = 0|\mathbf{x}) + p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x})}{1 + p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x})}}, \\ &= \frac{(1 - p(y = 1|\mathbf{x})) \cdot (1 + p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x}))}{(1 - p(y = 1|\mathbf{x})) + p(y = 1|\mathbf{x}) \cdot p(h_v > e_v | y = 1, \mathbf{x})}, \end{aligned} \quad (18)$$

Using the binary cross-entropy loss and substituting the importance weights, we obtain the final loss:

$$\mathcal{L}_{\text{ES-DFM}} = - \sum_{(\mathbf{x}, y) \in \mathcal{P}_v \cup \mathcal{N}_v} [y \cdot w_v^+ \cdot \log \hat{p}_v(\mathbf{x}) + (1 - y) \cdot w_v^- \cdot \log(1 - \hat{p}_v(\mathbf{x}))], \quad (19)$$

## D Experiments

### D.1 Data Settings

To simulate continuous training, we split our dataset into two parts. The first part is for pre-training to provide a strong initial setup. We put a special gap between the two parts based on conversion and refund attribution windows ( $W_v^{\text{attr}} + W_r^{\text{attr}}$ ) to prevent label leakage. The second part is used to train and evaluate the models and is further divided into 0.01-day (15-minute) segments. Following the online learning approach [4, 8], we train on the  $n$ -th segment and evaluate on the **clicks of the  $(n + 1)$ -th segment**, with metrics averaged across all segments.

We define four types of windows for **CASCADE**: conversion attribution  $W_v^{\text{attr}}$ , refund attribution  $W_r^{\text{attr}}$ , conversion observation  $W_v^{\text{obs}}$ , and refund observation  $W_r^{\text{obs}}$ . The conversion and refund

**Table 6: Statistics of the datasets. Counts are rounded in millions (M).**

| Dataset | #Users | #Items | #Clicks | #Conversions | CVR   | #Refunds | Duration |
|---------|--------|--------|---------|--------------|-------|----------|----------|
| CASCADE | 2.0M   | 3.1M   | 41.0M   | 3.7M         | 0.091 | 2.0M     | 25 days  |

attribution windows are both set to 3 days, defining the maximum delay for valid outcomes, while observation windows are 0.01 day (15 minutes) each.

## D.2 Sample Delivery

Conversion events are fed into the model at the end of the conversion observation window if they occur within it; otherwise, they are added at the exact time of conversion. Refund events are fed into the model either at the time of actual refund or at the end of the refund observation window, whichever comes first. Notably, *the refund observation window is a unique feature of TESLA*. Other methods only detect refunds at the actual refund time.

## D.3 Detailed Evaluation Metrics

We adopt the following metrics to evaluate both ranking and calibration performance in streaming NetCVR prediction:

- **AUC**: The area under the ROC curve, measuring the model’s ability to distinguish between positive and negative samples in terms of pairwise ranking.
- **PRAUC**: The area under the precision-recall curve. Due to the highly imbalanced nature of streaming data, PRAUC is more sensitive than AUC to improvements on positive samples and better reflects practical utility.
- **NLL**: The negative log-likelihood, which evaluates the quality of predicted probabilities:

$$\text{NLL} = -\frac{1}{N} \sum_{i=1}^N [y_i \log \hat{p}_i + (1 - y_i) \log(1 - \hat{p}_i)],$$

It penalizes miscalibration and is critical in cost-per-action (CPA) systems.

- **RI-AUC** (Relative Improvement in AUC): Following prior works [4, 24], we compute the relative improvement over the pre-trained model:

$$\text{RI-AUC} = \frac{\text{AUC}_{\text{model}} - \text{AUC}_{\text{pre-trained}}}{\text{AUC}_{\text{oracle}} - \text{AUC}_{\text{pre-trained}}} \times 100\%,$$

where  $\text{AUC}_{\text{oracle}}$  is computed using ground-truth labels with full delay observation. RI-AUC measures how close a method approaches the oracle performance, with higher values indicating better recovery from delayed feedback.

- **PCOC** (Predicted Click-to-Conversion Over Calibration): We further evaluate calibration by computing the ratio of average predicted probability to empirical conversion rate:

$$\text{PCOC} = \frac{\mathbb{E}[\hat{p}]}{\mathbb{E}[y]},$$

A PCOC value close to 1 indicates well-calibrated predictions. Values  $> 1$  indicate overestimation (common in delayed feedback), while  $< 1$  indicate underestimation.

## D.4 Detailed Competitors

The compared methods include:

- **Pre-trained**: The initial model trained on the pre-training set without any online updates. Serves as a static baseline to measure the value of streaming adaptation.
- **Oracle**: A privileged model fine-tuned with ground-truth labels after full observation (i.e., after  $W_c + W_r$ ), representing the performance upper bound under complete feedback.
- **BDL (Batch Deep Learning)**: A daily batch model that waits for all feedback within attribution window, thus receiving delay-free labels. Simulates ideal offline training with no label censorship.
- **FNC** [11]: Corrects selection bias by directly re-injecting delayed feedback samples into the training stream upon their arrival.
- **FNW** [11]: Uses importance weighting to up-weight fake negatives and down-weight real negatives based on delay distribution.
- **DEFER** [8]: Replicates both delayed positive samples and confirmed real negative samples to mitigate feature distribution shift caused by asymmetric sample duplication.
- **DEFUSE** [4]: Introduces a four-way categorization of samples: Immediate Positive (IP), Fake Negative (FN), Real Negative (RN), and Delayed Positive (DP), with tailored modeling and reweighting for each type.
- **ES-DFM** [24]: The first method to introduce an explicit conversion observation window and perform debiasing within it using importance weighting. Models the trade-off between label completeness and data freshness via adaptive sample reweighting.
- **DDFM** [6]: Proposes a hybrid modeling strategy: applies unbiased estimation for samples inside the observation window, and debiased modeling for those outside—without relying on survival models.
- **DSFN** [14]: Leverages transfer learning to capture shared representations across different types of streaming user feedback.
- **MISS** [13]: Employs multiple models trained on different observation windows and fuses their predictions via ensemble voting to improve robustness.

## D.5 Implement Details

We reproduce all competitors based on their official implementations. Since cascaded modeling outperforms direct modeling (§ 3.3.2), all methods adopt a unified dual-tower architecture that independently estimates CVR and RFR. All models are initialized from the same pre-trained checkpoint and fine-tuned on identical streaming data, ensuring controlled comparisons under cascaded feedback. We implement **TESLA** and all baselines in PyTorch. Feature inputs are embedded and fed into fully connected networks with hidden sizes {256, 256, 128}, where each layer is followed by BatchNorm [10] and LeakyReLU [18]. All methods are optimized with Adam [15], and hyperparameters are carefully tuned to report the best performance.

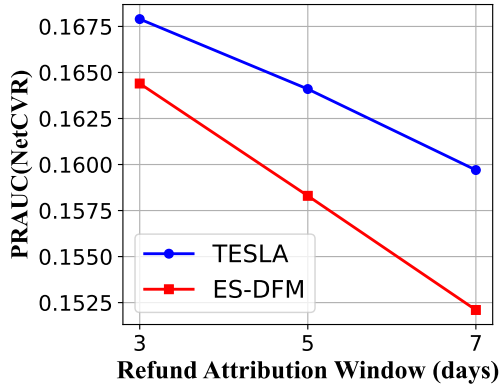


Figure 9: Robustness study on refund attribution window.

## D.6 Robustness to Refund Attribution Windows

To evaluate robustness under different refund attribution assumptions, we further test **TESLA** with varying refund attribution windows  $W_r^{attr} \in \{3d, 5d, 7d\}$  while keeping all other settings fixed. This design intentionally introduces variation in the time allowed for attributing refunds, thereby simulating the practical uncertainty in real-world refund timing. It allows us to examine whether the performance is sensitive to such shifts in the attribution window.

The results indicate that **TESLA** consistently outperforms strong delayed-feedback baselines across all experimental settings. As illustrated in Fig. 9, the proposed **TESLA** consistently outperforms ES-DFM across different refund attribution windows in terms of NetCVR PR-AUC, with relative improvements of approximately 2.1%, 3.7%, and 5.0% for the 3-day, 5-day, and 7-day windows, respectively. Importantly, the performance gap remains stable as the refund window increases, suggesting that **TESLA is robust to variations in refund timing** and does not depend on finely tuned attribution assumptions. These findings highlight **TESLA's effectiveness in handling realistic refund delay shifts**.