Modelling Delayed Redemption with Importance Sampling and Pre-Redemption Engagement

Samik Datta
Amazon
samdatta@amazon.com

Anshuman Mourya
Amazon
mouryaan@amazon.com

Anirban Majumder
Amazon
majumda@amazon.com

Vineet Chaoji
Amazon
vchaoji@amazon.com

ABSTRACT

Rewards-based programs are popular within e-commerce online stores, with the goal of providing serendipitous incentives to delight customers. These rewards (or incentives) could be in the form of cashback, free-shipping or discount coupons on purchases within specific categories. The success of such programs relies on their ability to identify relevant rewards for customers, from a wide variety of incentives available on the online store. Estimating the likelihood of a customer redeeming an incentive is challenging due to 1) *data sparsity*: relatively rare occurrence of coupon redemptions as compared to issuances, and 2) *delayed feedback*: customers taking time to redeem, resulting in inaccurate model refresh, compounded by data drift due to new customers and coupons.

To overcome these challenges, we present a novel framework, Dress (**D**elayed **R**edemption **E**ntire **S**pace **S**ampling), that jointly models the effect of data sparsity and delayed feedback on redemptions. Our solution entails an architecture based on the recently proposed Entire Space Model ([12]), where we leverage pre-redemption engagement of customers (e.g. clipping of coupon) to overcome the sparsity challenge. The effect of delayed feedback is mitigated via a novel importance sampling mechanism, whose efficacy we formally analyze via a novel application of Influence Function ([10]). Experimental evaluation suggests that Dress achieves significant lift in offline metric in comparison to state-of-the-art alternatives. Additionally, a live A/B test with Dress resulted in a lift of 10 basis points in the redemption rate.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; • **Mathematics of computing** → **Probabilistic inference problems**; • **Information systems** → *Display advertising*.

KEYWORDS

Delayed Feedback; Importance Sample; Influence Function; Entire Space Model

KDD '23, August 6–10, 2023, Long Beach, CA, USA

ACM ISBN 979-8-4007-0103-0/23/08.
https://doi.org/10.1145/3580305.3599867

ACM Reference Format:
Samik Datta, Anshuman Mourya, Anirban Majumder, and Vineet Chaoji. 2023. Modelling Delayed Redemption with Importance Sampling and Pre-Redemption Engagement. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3580305.3599867

1 INTRODUCTION

Identifying and surfacing relevant items, typically from a large set of items, is a fundamental building block for multiple online businesses. For instance, within e-commerce [13, 17, 21], surfacing relevant products within search or recommending pertinent products is critical for ensuring good customer experience. Similarly, relevance of an ad to a query within advertising [7, 14] determines a customer's click or the relevance of a post within a social network's feed [1, 5] impacts the user's engagement with it. News feeds [24] and video streaming platforms [16] are but a few other domains where relevant content plays a significant role towards business outcomes.

In this paper, we focus on the problem of coupon recommendation to customers. On e-commerce online stores, customers are issued rewards in the form of coupons that can either credit an instant cashback or offer a discount (*"Do X get Y"*), to be redeemed with the next transaction. These coupons serve the purpose of incentivizing customers to complete a purchase, which was perhaps not as affordable without the discount associated with the coupon. As a result, the recommender's goal is to maximize coupon redemptions, thereby maximizing purchases, through relevant coupon recommendations. Coupons are initially issued to customers on completion of certain actions on the online store. Prior to the actual redemption, customers typically engage with the online store, thereby leaving evidence in the form of pre-conversion actions. For instance, customers may inspect the coupon, optionally *collect* the coupon, and finally *redeem* the coupon by completing the purchase. Figure 1 captures transitions between user actions. The labels on the edges represent a) the probability of moving from one action to another, and b) mean delay (in hours) to transition to the next action. A sequence of actions, such as "click → add → purchase", results in a *trace*.

Estimating the likelihood of redemption, within the context of coupons, is a challenging problem. First, due to the relatively rare occurrence of redemptions, the number of training samples available across all possible user traces is small. Coupled with the industry trend of employing models with a large number of parameters, the

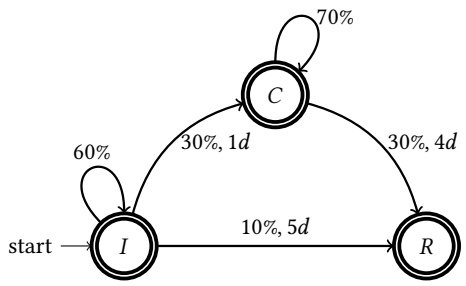

Figure 1: Transition diagram for coupon states: issued (I), clipped (C) and redeemed (R). The edge labels show data sparsity (in percentage of positives) and median delay (in days). Figure is only for illustration purpose and actual numbers are not revealed due to confidentiality.

obtained samples are insufficient to fit the large parameter space of the model. This *data sparsity* problem can manifest in inaccurate and over-confident estimation of redemption rates. On the other hand, arrival of new customers and coupon types on the online store, and varying redemption rates through the month, result in changes in the data distribution. To keep the model up-to-date, this necessitates frequent retraining of models deployed in a production system.

Additionally, redemption is not an instantaneous event as customers take time to decide the product they want to purchase. Publicly available data [1] from a popular e-commerce search engine shows that the median time between click and conversion varies anywhere between couple of hours to more than a week, across different product categories. This leads us to the *delayed feedback* problem which introduces a dilemma in model training - on one hand, it is pragmatic to wait long enough such that the observed distribution of redemption closely matches with the true distribution; on the other hand, the model needs to be retrained frequently to deal with the evolving data distribution.

Recent works have addressed these challenges separately, in a siloed fashion, primarily in the context of display advertising. For example, Ma et al. [12] proposes an architecture (Entire Space Model or ESM) that models conversion prediction as a multi-task problem, by simultaneously predicting clicks and conversions. Wen et al. [23] extend this work by decomposing conversion estimation into multiple post-click behavioral signals and composing the loss terms based on sum-product rules of probability. However, unlike the "impression → click → conversion" trace commonly assumed in advertising, many domains have other pre-conversion signals such as "add-to-cart", "add-to-wishlist", etc. which are not mandatory prerequisites for a conversion. For example, a customer may purchase an item without any click or add-to-cart, probably swayed by a deep discount, corresponding to the trace "impression → purchase". Hence the general conversion estimation problem warrants a non-trivial departure from the ESM architecture.

There is an extensive literature on modeling delayed feedback in the context of advertising, starting with the seminal work of Chapelle [3] that models the joint probability distribution over conversion and the time delay between the click and the conversion events. Recent research [25] has focused on a sampling based approach that models the relationship between observed and true distribution of conversion and subsequently optimizes the loss function under the true distribution via importance sampling mechanism. However, there are few practical constraints ignored by contemporary literature. First of all, items often have a finite lifetime after which the observed negatives become true negatives. In the case of coupons, once the coupon expires after a pre-specified duration, the issuance can be counted as a true negative. Further, the redemption model uses historical redemption rate as a feature, for which computation is also affected due to delayed feedback. Traditional work in this area completely ignores the uncertainty in the feature vector distribution due to the delayed nature of the target variable. Further, there has been no prior work on addressing data sparsity and delayed feedback jointly.

In this work, we present Dress (**D**elayed **R**edemption **E**ntire **S**pace **S**ampling), a unified architecture for coupon redemption rate estimation to address the challenges of data sparsity and delayed feedback of the redemption signal. Note that Dress is agnostic of the underlying learning algorithm for the redemption model, instead acts as a wrapper (i.e., "dress") around the core model.

Specifically, we make the following contribution in this paper,

(1) We introduce the problem of coupon ranking, in the context of providing rewards to customers. We highlight some unique challenges that warrant non-trivial solutions.
(2) To the best of our knowledge, Dress is the first work that jointly models data sparsity and delayed feedback of target labels in the context of personalized coupon relevance estimation.
(3) We present a theoretical framework to quantify the efficacy of Dress, through the lens of Influence Function [10].
(4) We conduct experiments on multiple datasets, including publicly available datasets to highlight the performance of our solution in comparison to a wide range of baselines. Experimental evidence suggests that our solution achieves significant lift in offline metrics as compared to state-of-the-art approaches.
(5) Dress is deployed in production for an online comparison with the incumbent strategy and results in an improvement of +10 basis points in redemption rates, a non-trivial improvement in terms of business metrics.

The rest of the paper is organized as follows. In Section 2, we introduce notations and formally define the business problem, followed by the components of Dress, our key technical contribution. Section 3 presents the theoretical framework for analysing Dress, while Section 4 presents a thorough and comprehensive set of experiments.

2 DELAYED REDEMPTION ENTIRE SPACE SAMPLING

We present Dress, a novel framework to jointly model the effect of data sparsity and delayed feedback of redemptions. Our solution entails a novel architecture based on recently proposed Entire Space Model (ESM [12]) where we leverage post-issuance activity of users (e.g. clipping of coupons) to overcome the sparsity challenge. The effect of delayed feedback is mitigated via a novel importance sampling mechanism. First, we introduce some notation.

[1] https://tianchi.aliyun.com/dataset/649

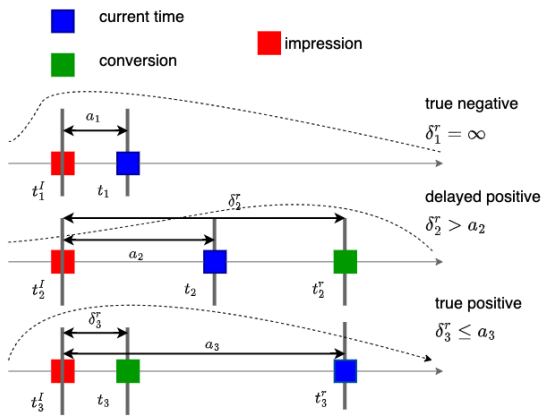

Figure 2: An illustration of true negatives, delayed positives and true positives.

Notation. We let $x_i \in \mathbb{R}^p$ denote the feature vector corresponding to the i^{th} coupon issuance that combines static attributes of coupon (discount, GL etc.), customer (e.g. demographics) and behavioral features mined from past interactions on the online store. We use $y_i^c \in \{0, 1\}$ (and y_i^r) to denote the collect (redemption resp.) event and t_i^c, t_i^r to denote the corresponding timestamps. Further, let t_i^I, t_i^e be the issuance and expiry times. The age of an issuance is defined as $a_i = t - t_i^I$ where t is the current time. The timestamps are used to define corresponding time intervals: $\delta_i^c = t_i^c - t_i^I, \delta_i^r = t_i^r - t_i^I, \delta_i^e = t_i^e - t_i^I$. To avoid notational clutter, we drop the suffix i whenever the context is clear. There are three possibilities of the target label [2]: (1) positives: $(y^r = 1)\&(\delta^r \leq a)$ (2) false negatives (or delayed positives): $(y^r = 1)\&(\delta^r > a)$, (3) true negatives: $(y^r = 0)\&(\delta^e \leq a)$. Note that for true negatives, $\delta^r = \infty$.

2.1 Delayed Feedback Model for Redemption

We first describe the construction of a delayed feedback model for redemption, although the techniques presented here can be applied for clip prediction with minimal changes. The training dataset $\mathcal{D}_t$ at time t consists of samples of the form $\{(x_i, y_i^r, a_i, \delta_i^r)_i\}$ where the label y_i^r is defined based on values observed till time t. If there was no delay i.e. $a_i \geq \delta_i^r$ for all training instances $i \in \mathcal{D}_t$, then one can fit a predictive model $y^r = f_r(x; \theta_r)$ by minimizing the loss $\mathcal{L} = \mathbb{E}_{(x,y^r)\sim p(x,y^r)} \ell(y^r, f_r(x; \theta_r))$ where $p(x, y^r)$ is the data generating distribution. However, because of delayed redemptions, we don't have access to $p(\cdot)$. Instead we believe the samples in $\mathcal{D}_t$ are drawn from an observed distribution $q(x, y^r)$ (different from $p(\cdot)$) and we venture to minimize $\mathcal{L} = \mathbb{E}_{(x,y^r)\sim q(x,y^r)} \alpha(x, y^r) \cdot \ell(y^r, f_r(x; \theta_r))$, where the importance weight term $\alpha(x, y^r)$ is defined as

$$\alpha(x, y^r) = \frac{p(x, y^r)}{q(x, y^r)} = \underbrace{\left(\frac{p(y^r \mid x)}{q(y^r \mid x)}\right)}_{\text{label correction}} \cdot \underbrace{\left(\frac{p(x)}{q(x)}\right)}_{\text{feature correction}} \tag{1}$$

[2]Here we talk about redemptions but the same applies to clip actions as well.

Note that delay correction with importance sampling involves two factors corresponding to the target labels and the features. It is common for many production systems to use historical event rate (e.g. historical CTR) as warm-start features in the model whose computation is biased due to delayed arrival of the target label. Unlike prior works that ignore the feature computation bias, our framework allows us to seamlessly incorporate both the effects.

2.1.1 *Importance Sampling for Label Bias Correction.*

We train a wait-time prediction model $p(w^r \mid x)$ that tells us how long we must wait before the observed redemption label for issuance x can be consumed (for model training purpose). Note that the wait time construct gives us a controllable lever to trade-off freshness of the model with maturity of the target labels. It is imperative to partition training dataset $\mathcal{D}_t$ into three sets after sampling a wait time $w_i^r \sim p(w^r \mid x_i)$ for each issuance: (1) **eden generation** ($\mathcal{D}_t^e$): contains instances from $\mathcal{D}_t$ where $a_i < w_i^r$. Instances in the eden generation are excluded from the training dataset as they are deemed to be immature. (2) **old generation** ($\mathcal{D}_t^o$): issuances from the past where the enclosed coupon has expired i.e. $\delta_i^e \geq a_i$, thus lending the feedback observable. For instances in $\mathcal{D}_t^o$, $p(x, y^r) = q(x, y^r)$; (3) **young generation** ($\mathcal{D}_t^y = \mathcal{D}_t \setminus (\mathcal{D}_t^e \cup \mathcal{D}_t^o)$): where the age a_i of an instance is more than the wait time w_i^r. These are the only issuances with $p(x, y^r) \neq q(x, y^r)$.

We begin by deriving the relationship between $q(y^r|x)$ and $p(y^r|x)$ which simplifies the derivation of importance weights.

$$q(y^r = 1|x) = \underbrace{p(y^r = 1|x) \cdot p(\delta^r \leq a|x, y^r = 1)}_{\text{Early Positive}} \tag{2}$$

$$q(y^r = 0|x) = \underbrace{p(y^r = 0|x)}_{\text{Real Negative}} + \underbrace{p(y^r = 1|x) \cdot p(\delta^r > a|x, y^r = 1)}_{\text{Delayed Positive}} \tag{3}$$

We plug in our estimate of $q(\cdot)$ to derive a closed-form expression for the importance weights (Equation 1):

$$\frac{p(y^r = 0|x)}{q(y^r = 0|x)} = \frac{p(y^r = 0|x)}{p(y^r = 0|x) + p(y^r = 1|x) \cdot p(\delta^r > a|x, y^r = 1)} := g_{tn}^r(x, a) \tag{4}$$

$$\frac{p(y^r = 1|x)}{q(y^r = 1|x)} = \frac{1}{p(\delta^r \leq a|x, y^r = 1)} := \frac{1}{1 - S^r(x, a)} \tag{5}$$

Here we introduce a pair of auxiliary functions: a classifier $g_{tn}^r(x, a)$ that estimates the likelihood of an issuance being true negative after having observed no redemption till the point of data collection and a survival function, $S^r(x, a)$, that models the delay in arrival of the redemption signal. Note that to avoid large values of the survival function in Equation 5 (and hence leading to division by zero), we override the sampled wait time as $w^r \leftarrow \max(w^r, \beta)$ where β is a percentile of wait time distribution.

2.1.2 *Importance Sampling for Feature Bias Correction.*

Armed with importance weights for labels, we now turn our attention to deriving importance weights $\frac{p(x)}{q(x)}$ to account for uncertainty in the feature vector. Note that this correction step is required only for issuances in $\mathcal{D}_t^y$ and limited to features that directly depend on the target label (e.g. historical redemption rate of a coupon) Assuming that $p(x)$ and $q(x)$ are factorizable, we have:

$$\frac{p(x)}{q(x)} = \prod_{k \in \mathcal{F}} \frac{p(x_k)}{q(x_k)}, \forall x \in \mathcal{D}_t^y \tag{6}$$

$$\frac{p(x_k)}{q(x_k)} = \prod_{i \in \mathcal{I}_k} \frac{p(y_i^r|x_i)}{q(y_i^r|x_i)} = \prod_{i \in \mathcal{I}_k^+} \frac{1}{1 - S^r(x_i, a_i)} \times \prod_{i \in \mathcal{I}_k^-} g^r(x_i, a_i) \tag{7}$$

Here $\mathcal{F}$ contains the indices of features requiring the delay correction, $\mathcal{I}_k$ denotes past issuances that appear in the computation of the feature x_k, $\mathcal{I}_k^+ := \{i \in \mathcal{I} | y_i^c = 1\}$, and $\mathcal{I}_k^- := \mathcal{I}_k \setminus \mathcal{I}_k^+$.

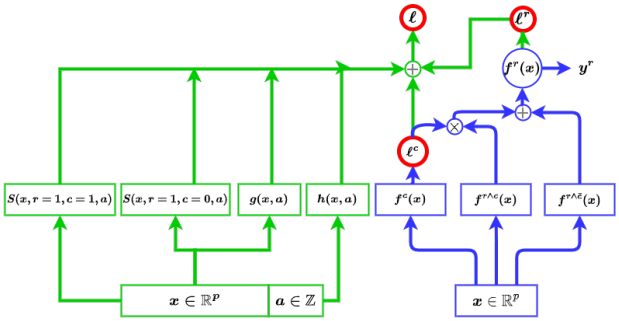

Figure 3: High-level architecture of DRESS including the training (left, colored in green) and inference (right, colored in blue) stages.

2.2 Entire Space Model for Redemption

Entire Space Model (ESM) is built upon the simple principle of sum-product rule in probability. In our specific context, a redemption may occur with or without a prior clip, thus leading to

$$\begin{aligned} p(y^r = 1|x) = p(y^r = 1|y^c = 1, x) \cdot p(y^c = 1|x) \\ + p(y^r = 1|y^c = 0, x) \cdot p(y^c = 0|x) \end{aligned} \tag{8}$$

Thus, to build a predictive model for redemption, one may begin with modelling each of the constituent terms: $p(y^r = 1|y^c = 1, x) := f^{r \wedge c}(x)$, $p(y^c = 1|x) := f^c(x)$, and, $p(y^r = 1|y^c = 0, x) \cdot p(y^c = 0|x) = p(y^r = 1, y^c = 0|x) := f^{r \wedge \bar{c}}(x)$. At a high-level, the network computes individual terms $f^{r \wedge c}(x)$, $f^c(x)$, $f^{r \wedge \bar{c}}(x)$ and applies the sum-product rule (Equation 8) to estimate $p(y^r = 1) = f^r(x) = f^c(x) \cdot f^{r \wedge c}(x) + (1 - f^c(x)) \cdot f^{r \wedge \bar{c}}(x)$. The model is trained by minimizing the loss over the predicted clip and redemption,

$$\mathbb{E}_{(x, y^c, y^r) \sim p(x, y^c, y^r)} \left\{ \ell(y^c, f^c(x)) + \ell(y^r, f^r(x)) \right\} \tag{9}$$

2.3 DRESS : Delayed Feedback Entire Space Model for Redemption

We are now in a position to illustrate how the building blocks developed thus far are composed to achieve our final objective: an entire space model that can handle delayed feedback in both clip and redemption. Training involves minimization of the importance-weighted loss function,

$$\mathbb{E}_{(x, y^c, y^r) \sim q(x, y^c, y^r)} \alpha(x, y^c, y^r) \times \left\{ \ell^c(y^c, f^c(x)) + \ell^r(y^r, f^r(x)) \right\}$$

The form of the importance weights depends on the exact values of the labels y^c, y^r which we derive next.

Case I ($y^r = 1, y^c = 1$). As both clip and redemption are observed, we can write $q(y^r = 1, y^c = 1 \mid x) = p(y^r = 1, y^c = 1, \delta^r \le a, \delta^c \le a)$. The importance weight in this case can be written as

$$\alpha(x, y^c = 1, y^r = 1) = \frac{1}{p(\delta^r \le a, \delta^c \le a \mid y^c = 1, y^r = 1, x)} \tag{10}$$

Case II ($y^r = 1, y^c = 0$). As clip doesn't follow redemption, $y^c = 0$ is a true negative. The observed distribution can be written as $q(y^r = 1, y^c = 0 \mid x) = p(y^r = 1, y^c = 0, \delta^r \le a)$. The importance weight in this case can be written as,

$$\alpha(x, y^c = 0, y^r = 1) = \frac{1}{p(\delta^r \le a \mid x, y^r = 1, y^c = 0)} \tag{11}$$

Case III ($y^r = 0, y^c = 1$). This scenario encompasses two possibilities: true-negative redemption and delayed-positive redemption. The observed probability is the net sum of these two outcomes. The importance weight $\alpha(x, y^c = 1, y^r = 0)$ can be written as the following ratio (for brevity, detailed derivation is deferred to Appendix A),

$$\frac{p(y^c = 1, y^r = 0 \mid x)}{p(y^r = 0, y^c = 1, \delta^c \le a \mid x) + p(y^r = 1, y^c = 1, \delta^c \le a, \delta^r > a \mid x)} \tag{12}$$

Note that we don't need to estimate individual terms, the importance weight can be estimated by a **conditional true negative classifier** $g(x, a) = p(y^r = 0 \mid x, y^c = 1, \delta^c \le a)$. The classifier is trained on all samples with observed clip signal, i.e. $y^c = 1$ and $\delta^c \le a$.

Case IV ($y^r = 0, y^c = 0$). We explore all four possibilities for the observed labels corresponding to true-negative and false-negative clip and redemption. The importance weight in this case can be estimated by a **joint true negative classifier** $h(x, a) = p(y^r = 0, y^c = 0 \mid x, \delta^r > a, \delta^c > a)$. The classifier is trained on samples with no observed clip and redemption up to time $t = t_I + a$. For brevity, detailed derivation is deferred to Appendix A.

The estimator for importance weights can be plugged in to derive the correction terms for features as shown in Section 2.1.2. Figure 3 shows the high-level architecture of DRESS. All components are implemented as deep neural networks with learnable set of parameters. The right part of the figure (colored in blue) implements the entire space model which is queried during the inference time. The left part (colored in green) is used only during training and implements classifiers and survival functions to estimate the importance scores.

3 RISK MINIMISATION IN DRESS

We now develop a theoretical framework to analyze the efficacy of DRESS, as well as the related family of algorithms based on Importance Sampling (IS). The framework is built upon Influence Function [10], which studies the effect of training dataset perturbation on model parameter and test risk.

3.1 Background

Had we enjoyed access to $\mathcal{D}_\infty$ – the *a*ged version of $\mathcal{D}_t$, where $a_i \ge \delta_i^r, \forall i \in [N]$, so that y_i^r reflects the *e*ventual redemption label – we

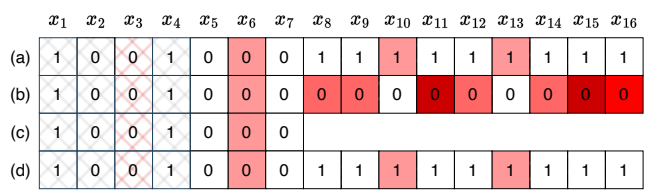

Figure 4: Effect of Flip and Imps through the lens of Influence Function. Row (a) corresponds to $\mathcal{D}_\infty$, with samples $x_1, \cdots, x_4$ (hatched) in the Old Generation, and $x_5, \cdots, x_{16}$ in the Young Generation. The content of the cell is its label, and the color indicates the value of Φ_i°: *benign*, $\Phi_i^\circ < 0$, is painted white, whereas *harmful*, $\Phi_i^\circ > 0$, is painted red. A darker shade of red implies accentuated harmfulness (we omit the color gradient on benign samples for clarity). Row (b) corresponds to $\mathcal{D}_t$, which is the result of Flip on (a). Row (c) and (d) result from applying Imps and Flip on (b), respectively.

could obtain the optimal parameter under the Empirical Risk Minimisation framework, $\widehat{\theta} \coloneqq \arg\min_\theta \sum_{i=1}^N \frac{1}{N}\ell(y_i^r, f_r(x_i;\theta))$. Similarly, given access to a set of M test samples, $\mathcal{D}_\infty^M \coloneqq \{(x_j, y_j^r)\}$ drawn i.i.d. from $p(x, y^r)$, we could calculate the *e*mpirical test risk as $\mathcal{L}^M(\widehat{\theta}) \coloneqq \frac{1}{M}\sum_{j=1}^M \ell(y_j^r, f_r(x_j;\widehat{\theta}))$.

In what follows, we study the perturbation in $\mathcal{L}^M(\widehat{\theta})$ resulting from a perturbation of the training dataset, $\mathcal{D}_\infty$, which, in turn, perturbs $\widehat{\theta}$. In particular, we study two types of perturbations: Flip, where redemption labels are selectively flipped, $y_i^r \mapsto z_i^r$, inducing a corresponding change in the associated training loss, $\frac{1}{N}\ell(y_i^r, f_r(x_i;\theta)) \mapsto \frac{1}{N}\ell(z_i^r, f_r(x_i;\theta))$; and, Imps, where each training loss term is scaled in accordance with IS: $\frac{1}{N}\ell(y_i^r, f_r(x_i;\theta)) \mapsto \frac{1}{N}\frac{p(x_i,y_i^r)}{q(x_i,y_i^r)}\ell(y_i^r, f_r(x_i;\theta))$. Specifically, setting $\frac{p(x_i,y_i^r)}{q(x_i,y_i^r)} = 0$ amounts to discarding sample i from the training dataset.

In order to quantify the effect of perturbation on empirical test risk, Influence Function studies an interpolated version of the perturbation, where an infinitesimal change is administered to the training dataset. The interpolated version of Flip is: $\frac{1}{N}\ell(y_i^r, f_r(x_i;\theta)) \mapsto \frac{1}{N}\ell(y_i^r, f_r(x_i;\theta)) + \epsilon_i\ell(z_i^r, f_r(x_i;\theta)) - \epsilon_i\ell(y_i^r, f_r(x_i;\theta))$, where $\epsilon_i \in [0, \frac{1}{N}]$ is the amount of the change, with $\epsilon_i = \frac{1}{N}$ signifying the completion of the Flip operation. Similarly, the interpolated version of Imps is: $\frac{1}{N}\ell(y_i^r, f_r(x_i;\theta)) \mapsto \frac{1}{N}\ell(y_i^r, f_r(x_i;\theta)) + \epsilon_i\ell(y_i^r, f_r(x_i;\theta))$, where $\epsilon_i \in \left[0, \frac{1}{N}\left(\frac{p(x_i,y_i^r)}{q(x_i,y_i^r)} - 1\right)\right]$ is the amount of the change.

Infinitesimal perturbation allows one to quantify the corresponding rate of change in empirical test loss, $\Phi_i^\circ \coloneqq \frac{\partial \mathcal{L}^M(\widehat{\theta_{\epsilon_i}})}{\partial \epsilon_i}\Big|_{\epsilon_i=0}$, where $\widehat{\theta_{\epsilon_i}}$ is the optimal model parameter obtained by minimising the perturbed training loss, and $\circ$ denotes the specific perturbation operator (or a composition thereof). To this end, [11] furnishes the following closed-form expression for Φ_i^{FL}, corresponding to Flip, when the loss function is twice-differentiable and strictly convex:

$$\Phi_i^{\mathsf{FL}} = -\nabla_\theta \mathcal{L}^M(\widehat{\theta}) H_{\widehat{\theta}}^{-1}\left(\nabla_\theta \ell(z_i^r, f_r(x_i;\widehat{\theta})) - \nabla_\theta \ell(y_i^r, f_r(x_i;\widehat{\theta}))\right) \tag{13}$$

Where, $H_{\widehat{\theta}} = \frac{1}{N}\sum_{i=1}^N \nabla_\theta^2 \ell(y_i^r, f_r(x_i;\widehat{\theta}))$ is the Hessian matrix computed at $\widehat{\theta}$. Similarly, [10] furnishes the following close-form expression for Φ_i^{IS}:

$$\Phi_i^{\mathsf{IS}} = -\nabla_\theta \mathcal{L}^M(\widehat{\theta}) H_{\widehat{\theta}}^{-1} \nabla_\theta \ell(y_i^r, f_r(x_i;\widehat{\theta})) \tag{14}$$

For a small, non-infinitesimal perturbation, ϵ_i, following [10], we approximate the resultant change in empirical test loss as $\epsilon_i \times \Phi_i^\circ$. Lastly, following [9], the net change caused by perturbing a *group* of training samples by ϵ_i each, $\forall i \in \mathcal{I}$, is approximated as $\sum_{i\in\mathcal{I}} \epsilon_i \times \Phi_i^\circ$. Equipped with this framework, we now turn our attention to the analysis of delayed redemption.

3.2 Risk Amplification

Delayed redemption can be viewed as an instance of the Flip operation on $\mathcal{D}_\infty$, where the (delayed) positive samples, with $y_i^r = 1, a_i < \delta_i^r$, are declared negative in $\mathcal{D}_t$ by setting $z_i^r = 0$. This operation introduces False Negative (FN) samples in $\mathcal{D}_t$. Assuming the specific loss function in use is Binary Cross-Entropy, and summoning Eq. 13, we derive the following theorem:

Theorem 3.1 (Risk Amplification). *Let $\mathcal{D}_t^{\mathsf{FN}} \coloneqq \left\{(x_i, z_i^r) | (x_i, y_i^r) \in \mathcal{D}_\infty, y_i^r = 1, a_i < \delta_i^r\right\}$ be the set of FN samples in $\mathcal{D}_t$. Assuming $\widehat{\theta}$ to be the optimal model parameter obtained from $\mathcal{D}_\infty$, and $\widehat{\theta^{\mathsf{FL}}}$ to be the optimal model parameter obtained from $\mathcal{D}_t$, we have $\mathcal{L}^M(\widehat{\theta^{\mathsf{FL}}}) - \mathcal{L}^M(\widehat{\theta}) = -\frac{1}{N}\sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \frac{\Phi_i^{\mathsf{IS}}}{1 - f_r(x_i;\widehat{\theta})}$*

Discussion. Proof is deferred to Appendix B. For *harmful* [3] samples in $\mathcal{D}_t^{\mathsf{FN}}$ with $\Phi_i^{\mathsf{IS}} > 0$, Theorem 3.1 suggests a *reduction* in empirical test risk. On the other hand, the originally *benign* samples with $\Phi_i^{\mathsf{IS}} < 0$ have now been turned harmful by Flip. Moreover, since $f_r(x_i;\widehat{\theta}) \approx 1, \forall i \in \mathcal{D}_t^{\mathsf{FN}}$, the harmfulness is accentuated, resulting in an amplification in the empirical test risk by a factor of $\frac{1}{1 - f_r(x_i;\widehat{\theta})}$. Fig. 4 illustrates the process pictorially, where row (b) depicts the result of the Flip operation on row (a) highlighting the risk amplification. The left panel in Fig. 6 indeed demonstrates the amplified test risk demonstrated by Ftt+Eager, which corresponds to $\widehat{\theta^{\mathsf{FL}}}$.

3.3 Risk Mitigation

A natural question to ask at this point is whether IS indeed mitigates the amplified empirical test risk, $\mathcal{L}^M(\widehat{\theta^{\mathsf{FL}}})$. Specifically, if $\widehat{\theta^{\mathsf{IS}\circ\mathsf{FL}}}$ is the optimal parameter obtained by minimizing the (scaled) loss function introduced in §2.1, is $\mathcal{L}^M(\widehat{\theta^{\mathsf{IS}\circ\mathsf{FL}}}) \leq \mathcal{L}^M(\widehat{\theta^{\mathsf{FL}}})$?

To this end, we recall from Eq. 4 that each sample $i \in \mathcal{D}_t$ with a negative label, $y_i^r = 0$, is assigned an importance weight of $g_{tn}^r(x_i, a_i)$. If this function is perfect, it effectively drops *all* the FN samples in $\mathcal{D}_t^{\mathsf{FN}}$ – including the harmful ones – thus mitigating the amplified risk. We formalize this intuition in Theorem 3.2 below:

Theorem 3.2 (Risk Mitigation). *If $g_{tn}^r(x_i, a_i) = 0, \forall i \in \mathcal{D}_t^{\mathsf{FN}}$, and $g_{tn}^r(x_i, a_i) = 1$ for all other negative samples in $\mathcal{D}_t \setminus \mathcal{D}_t^{\mathsf{FN}}$, $\mathcal{L}^M(\widehat{\theta^{\mathsf{IS}\circ\mathsf{FL}}}) - \mathcal{L}^M(\widehat{\theta^{\mathsf{FL}}}) \leq -\frac{1}{N}\sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \Phi_i^{\mathsf{IS}} + \sum_{i\in\mathcal{D}_t | y_i^r=1} \frac{1}{N}\frac{1-\beta}{\beta}\Phi_i^{\mathsf{IS}}$*

[3] As noted in [11], harmful samples comprise of noisy and out-of-distribution samples.

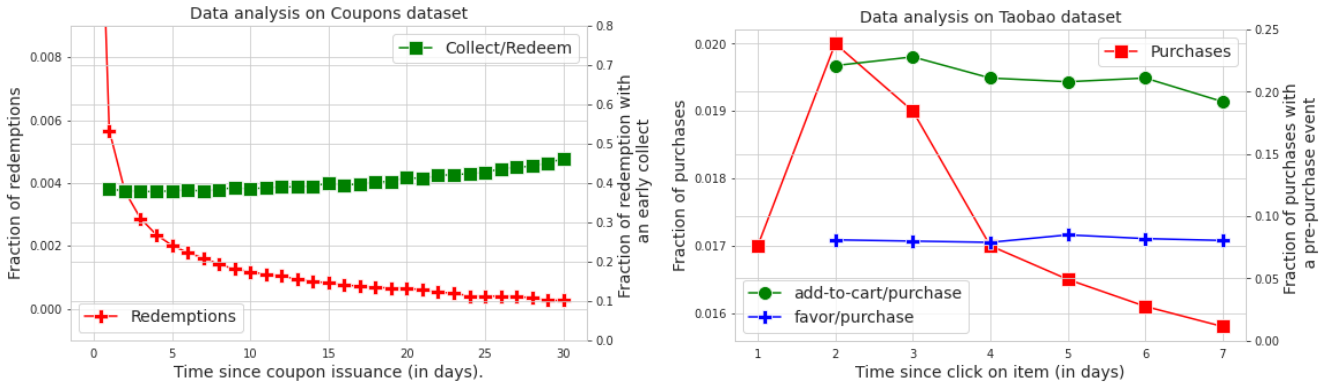

Figure 5: Left figure: a) fraction of redemptions (in red) taking place d days after coupon issuance ($d \in [0, 30]$), b) fraction of redemptions on day d that are preceded by a collect event on some day $d' < d$, on the Coupons dataset. Right figure: a) fraction of conversion on day d ($d \in [1, 7]$) after click on the item, b) fraction of conversions preceded by a pre-conversion signal (add-to-cart in green and favor in red) on some day $d' < d$.

Discussion. Proof is deferred to Appendix C. While IS has the desired effect on $\mathcal{D}_t^{\mathsf{FN}}$, it also perturbs the weight of the positive samples by a factor of $\frac{1}{1-S^r(x_i,a_i)}$ (refer to Eq. 5). As mentioned in §2.1.1, ensuring that we wait long enough (at least β percentile of the wait-time distribution) before inducting a training sample in the Young Generation, we have $1 - S^r(x_i, a_i) \geq \beta$. This further ensures that $\lim_{\beta \to 1} \mathcal{L}^M(\widehat{\theta^{\mathsf{IS} \circ \mathsf{FL}}}) - \mathcal{L}^M(\widehat{\theta^{\mathsf{FL}}}) = -\frac{1}{N} \sum_{i \in \mathcal{D}_t^{\mathsf{FN}}} \Phi_i^{\mathsf{IS}}$ – guaranteeing mitigation of the risk posed by $\mathcal{D}_t^{\mathsf{FN}}$ (recall that the Flip operation has turned originally-benign instances in $\mathcal{D}_t^{\mathsf{FN}}$ harmful [4], as established in Theorem 3.1). Row (c) in Fig. 4 illustrates this pictorially.

In reality, however, true-negative classifiers are not perfect. Specifically, when $g_{tn}^r(x_i, a_i) = \epsilon$ for all negative training samples, $\mathcal{D}_t^{\mathsf{N}}$, $\mathcal{L}^M(\widehat{\theta^{\mathsf{IS} \circ \mathsf{FL}}}) - \mathcal{L}^M(\widehat{\theta^{\mathsf{FL}}}) \approx \left(\epsilon - \frac{1}{N}\right) \sum_{i \in \mathcal{D}_t^{\mathsf{N}}} \Phi_i^{\mathsf{IS}}$. This, sadly, does not guarantee risk mitigation unless $\epsilon = o(\frac{1}{N})$, in which case, it effectively throws away *all* of $\mathcal{D}_t^{\mathsf{N}}$ – FN and TN alike.

Pre-redemption engagement signals incorporated in Dress boost the accuracy of the TN classifiers, $g(x, a)$ and $h(x, a)$, by adding informative features. Formally, Theorem 3 in [2] suggests that the test risk increases by an (additive) factor of $\sqrt{2I(Y^r; X^c | X^r)}$, where $I(Y^r; X^c | X^r)$ is the Conditional Mutual Information between the label, Y^r, and the features derived from pre-redemption engagement, X^c, given the rest of the features, X^r. The right panel in Fig. 6, where Mutual Information (MI) monotonically increases as more (synthetic) collect events are injected, demonstrates that test loss indeed decreases as MI increases.

4 EXPERIMENTS

In this section, we present a set of offline and online experiments to evaluate the performance of our model, in comparison to baseline techniques. We first present our findings on a coupons dataset collected from an emerging marketplace followed by results on a publicly available dataset on product recommendation. We glean some insights on the inner working of our model via a suite of semi-synthetic experiments. Finally, we conclude this section with findings from an online experiment.

4.1 Dataset

The Coupons dataset contains randomly sampled 25MM coupon issuances to customers, their redemptions and collect signals over a period of three months from an emerging marketplace. The objective is to estimate the redemption probability for each issuance. The customer data for the experiments with the Coupons data was anonymized and wherever possible numeric representations (aka. embeddings) were used. We also report our findings on a public recommendation dataset [26, 27] (referred to as Taobao). It contains user-item interactions from Taobao [5] during the period Nov 25 to Dec 3, 2017. Each record in the dataset is a user interaction of the form click (89.5%), purchase (2.1%), add-to-cart (5.5%) and favoring (2.9%). The dataset represents user shopping journey starting with a click on a product page which may eventually lead to a purchase with optional pre-purchase signals such as add-to-cart and favoring. We randomly sample 10MM interactions containing approximately 600k unique users and 2.4MM unique items. We group interactions by user-id and item-id, mapping purchase and pre-purchase signals to the nearest preceding click.

We conduct data analysis on both datasets to validate our assumption on Dress. In Figure 5 (left) we show the result of this analysis on the Coupons dataset. The figure shows that while a significant number of redemptions happen on the day of issuance, there is a long tail of redemptions leading up to a month. The green plot shows fraction of redemptions that are accompanied by an early collect i.e. that fraction of redemptions on day d that are preceded by a collect on day $d' < d$. It can be observed that this plot is quite stable ($\approx 10\%$ deviation) across days. Note that the y-axes in this plot are randomly scaled due to confidentiality.

Figure 5 (right) shows our analysis on the Taobao dataset. We plot fraction of purchases (in red) happening d days after the click on the item and fraction of conversions on day d with a prior add-to-cart (in green) and favor (in blue) action. While conversion

[4] For clarity of exposition, we overload Φ_i^{IS} to denote harmfulness in $\mathcal{D}_t^{\mathsf{FN}}$ – not in $\mathcal{D}_\infty$, unlike Theorem 3.1.

[5] https://tianchi.aliyun.com/dataset/649

Table 1: Comparison of offline performance of various delay strategies on the Coupons dataset in terms of ROCAUC and PRAUC metrics. Note that absolute numbers are not shown due to confidentiality. Instead we take Ftt + Eager as baseline and show relative improvements in metrics for other models. The models are evaluated with progressive validation strategy where metrics for only five randomly sampled days are shown for sake of brevity.

Models	day1		day2		day3		day4		day5	
	ROCAUC	PRAUC	ROCAUC	PRAUC	ROCAUC	PRAUC	ROCAUC	PRAUC	ROCAUC	PRAUC
Ftt + Lazy	-1.05%	-1.34%	-089%	-1.03%	-1.33%	-2.56%	-2.27%	-4.38%	-2.77%	-5.68%
Esm + Eager	-0.42%	0.34%	-0.76%	-1.67%	4.62%	21.04%	11.17%	8.75%	2.83%	22.16%
Ftt + Imps [25]	-0.68%	-1.17%	-0.10%	1.25%	4.99%	26.08%	12.73%	25.97%	**3.36%**	16.71%
Dress (Esm+ Imps)	-0.60%	**2.82%**	**0.92%**	**2.48%**	**6.19%**	**29.37%**	**12.98%**	**27.62%**	1.86%	**29.73%**

varies across days, there is little variation (< 5% variation) in the fraction of conversions with a pre-conversion signal. The following observation can be made from this experiment: a) there is large variation of redemption (conversion) across days which fools simple delay strategies such as Eager to incorporate wrong labels in model training, b) significant fraction of redemptions are accompanied by pre-redemption events. These pre-redemption signals are less sparse and they arrive sooner than the redemption label, which is our premise of Dress.

Table 2: Datasets used for evaluation of various delay strategies.

Dataset	**#samples**	**#users**	**#coupons /items**
Coupons	20MM	5MM	750
Taobao	10MM	0.6MM	2.4MM

4.2 Experimental Setup

We adopt the following evaluation protocols for offline evaluation: a) **out-of-time:** model is trained on 30 days of data and test set is selected as two weeks period subsequent to train, b) **progressive validation:** train on period $[d-30, d]$ and test set is selected on day $d+1$ and d is varied across the month. This protocol simulates typical industrial environment where models are retrained on a daily basis using latest available data. For model evaluation on the test set, we always use matured labels i.e. for an issuance on day d, the labels are borrowed from day $d+30$.

Featurization: For the coupons dataset, each issuance event is represented via a feature vector consisting of numeric attributes (incentive amount, historical redemption and collect rates, customer shopping history etc.) as well ID features (couponID, categoryID etc.). All numerical features are standardized with min-max scaling before feeding into the model. For categorical features, we learn their embeddings. The recommendation dataset doesn't provide any user feature. Therefore, for each click, we define the last 5 interacted (clicked, favored, added to cart and purchased) item-ids as a representation of user history. We use an embedding layer to learn dense representation of item-ids. To handle multiple pre-purchase signals, we take their union and the timestamp corresponding to the earliest event.

Survival Function & True-Negative Classifier. To simplify the design, in this work we do not implement the survival functions and real-negative classifiers as learnable models. Instead, we exploit the fact that although the collect and redemption time distributions vary across the coupon types, they are stable across days. In particular, we estimate the coupon type specific survival functions with the empirical univariate or bi-variate cumulative distribution functions (CDF). More formally, we represent the survival function as a table $\hat{F}(z, a)$ where z is the coupon type and each entry in the table represents $p(\delta^c \geq a, \delta^r \geq a \mid x, y^c = 1, y^r = 1) = \hat{F}(\text{type}(x), a)$. Here a represents the age of the coupon in unit of days. We follow similar procedure for representing the true-negative classifier empirically.

4.3 Baselines

For designing appropriate baselines, we first note that the majority of the literature, such as [25], that employ importance sampling to correct for delayed feedback has been developed in the context of streaming data, where training instances are considered immutable. Such baselines are inherently at a disadvantage in a batch setup, where can afford to utilize the most recent label information for each training instance. Furthermore, baselines suitable for a batch setup, such as [3], are not considered in our design space, given the iterative nature of the expectation-maximization algorithm which exacerbates the time and cost associated with each training run and they are already superseded by recent advancements (e.g. ES-DFM [25]).

We construct various baselines by incorporating the following delay strategies in model training: a) **Eager**: sets the target variable as per the observed redemption label. This strategy is commonly employed in practice due to its simplicity although it uses incorrect labels for delayed positives. b) **Lazy**: training data on day d consists of issuances till day $d-\Delta$, for a pre-specified parameter Δ. Note that this strategy trades off data freshness in favor of label correctness. For our experiments, we set $\Delta = 7$. c) **Imps**: importance sampling based scheme for delay correction as employed in Dress and in [25]. Note that Ftt+ Imps (or Mlp+ Imps) is same as a batch version of the popular ES-DFM [25] algorithm.

We use the recently proposed FTTransformer [6] (referred to as Ftt) as the baseline model in our experiments. Ftt learns a transformation of both numeric and categorical features via an embedding layer. It borrows the transformer architecture from NLP domain [20] and extends it to tabular datasets. Further, we report experiments on a two layer feed forward network (refered to as Mlp) to demonstrate that our algorithm gives performance boost across different choices of model architecture. Entire Space Model (referred to as Esm) is implemented by having separate heads for estimating

$p(y^c \mid x), p(y^r \mid y^c = 1, x), p(y^r \mid y^c = 0, x)$ and combining their output using sum-product probability rule to estimate $p(y^r \mid x)$. Depending on the context, we use either FTT or MLP as the backbone architecture for implementing ESM.

We use PyTorch [15] library to implement all our models. Training is done on a single Nvidia V100 GPU. Batch size is varied between [1024, 2048, 4096]. Adam optimizer is used for model training. The learning rate and weight decay parameters are tuned on a validation dataset and their optimal values are found to be $1e-4$ and $1e-5$ respectively. We employ early-stopping with patience set to 5. For FTT and MLP, we set embedding dimension to 128 with a dropout [18] of 0.1. Further, for FTT we use 3 layers of transformers and for MLP we use 2 layers of fully-connected networks with softmax activation, batchnorm [8] layer and dropout.

4.4 Results

4.4.1 Baseline Comparison. In Table 1, we show the offline performance of various delay strategies with the FTT baseline, in terms of ROCAUC and PRAUC metrics. For confidentiality reasons, we don't reveal the absolute numbers but present relative improvement w.r.t. the FTT + EAGER baseline (refer to Section 4.3). The models are evaluated using progressive validation strategy across all days in a month but in the table, we show metrics for randomly sampled 5 days, for sake of brevity. DRESS convincingly outperforms all the baselines across days achieving performance lift in the range of $0.92\% - 12.98\%$ on ROCAUC and $2.48\% - 29.73\%$ on PRAUC.

Table 3: Comparison of offline performance of various delay strategies using MLP architecture on the COUPONS dataset. MLP +EAGER is used as a baseline and relative improvements of models are reported. Models are evaluated with out-of-time evaluation strategy.

Models	ROCAUC	PRAUC
MLP + LAZY	-2.35%	-4.35%
ESM + EAGER	2.93%	**8.70%**
MLP + IMPS [25]	2.35%	0.00%
DRESS (ESM + IMPS)	**3.53%**	**8.70%**

We run further investigation to understand the variation in model performance across days. It is observed that coupons data distribution varies significantly across months due to change in coupon incentives, target segment of customers etc. At the start of the month, the training data primarily consists of issuances and redemptions from previous month which contributes to distribution drift between the train and test datasets. In presence of the drift, FTT+ EAGER being a simpler model, performs better than more sophisticated models such as DRESS. However, as the month progresses, training data consists of increasing number of samples from the current month and we see higher performance gains coming from DRESS and ESM+ EAGER.

The results further suggest that in a dynamic environment such as coupon recommendation, waiting for matured label (FTT+ LAZY) is a much inferior strategy than consuming incorrect labels (FTT+ EAGER). Further, leveraging any pre-redemption signal such as collect via ESM gives up to 22.16% lift as compared to using incorrect labels. In general, we observe that lifts are higher for PRAUC than ROCAUC. This is an artifact of class imbalance as ROC metric has been observed to be misleading for imbalanced datasets whereas, precision-recall metrics are specifically tailored for detecting rare events and are more reliable in those set-up. Table 3 presents offline metrics with the MLP architecture which shows similar trends.

In Table 4, we show the results of our offline experiments on the TAOBAO dataset. As this dataset has only 7 days of data, we evaluate using out-of-time strategy (first 5 days of data used for training). DRESS achieves significant improvement over all baselines in terms of ROCAUC and competitive performance on the PRAUC metric. Our hypothesis is that the performance of DRESS and all baselines on this dataset are limited due to a) absence of user long-term features, b) limited span of data that prevents us in accurately estimating the survival and real-negative classifiers.

Table 4: Comparison of offline performance of various delay strategies on the TAOBAO recommendation dataset using MLP architecture. Models are evaluated using out-of-time evaluation strategy.

Models	ROCAUC	PRAUC
MLP + EAGER	0.59	0.12
ESM + EAGER	0.64	**0.19**
MLP + IMPS [25]	0.62	0.15
DRESS (ESM + IMPS)	**0.68**	0.18

4.4.2 Semi-synthetic Experiments. To gain further insights on the performance of DRESS, we design a suite of semi-synthetic experiments on the COUPONS dataset. Our objective is to control a) the distribution of delay in redemption and collect signals and b) mutual information between redemptions and collects and observe how these choices impact the model performance. Following standard practice [3], we fit exponential distribution to the set of observed delays i.e. we model $p(\delta^r \mid \lambda_r) \propto \lambda_r \cdot e^{-\lambda_r \cdot \delta^r}$ and $p(\delta^c \mid \lambda_c) \propto \lambda_c \cdot e^{-\lambda_c \cdot \delta^c}$ where the parameters λ_r, λ_c are estimated via maximum likelihood estimation (MLE) technique. Let $\hat{\lambda}_r, \hat{\lambda}_c$ be the MLE estimates. We define $\alpha = \frac{\hat{\lambda}_c}{\hat{\lambda}_r}$.

In what follows, we vary $\frac{1}{\lambda_r} \in \{1, 10, 20 \cdots 100\}$ to simulate an environment where the mean delay in redemption is $\frac{1}{\lambda_r}$ and for each data-point (x, y^c, y^r) with $y^r = 1$, we sample a delay $\delta^r \sim p(\delta^r \mid \lambda_r)$. To account for delay in collects (for samples with $y^c = 1$), we set $\lambda_c = \lambda_r \cdot \alpha$ and sample $\delta^c \sim p(\delta^c \mid \lambda_c)$. Each simulation is repeated 5 times and we report the average over runs. Note that for this simulation, we manipulate only the delays and not the target labels. We also consider an *oracle* version where redemptions are known instantly, corresponding to delay of 0. We follow out-of-time validation strategy for all experiments.

In Figure 6 (left), we plot the test loss for DRESS and the baseline FTT model. Note that the baseline model uses EAGER strategy for delay correction. We observe that, in general, loss increases with higher delays for both the models. However, DRESS is robust to the variation in redemption delay incurring $< 5\%$ increase between $\delta^r = 0$ and 100. On the other hand, FTT + EAGER increases monotonically incurring +100% increase in loss as the mean redemption delay is varied between the extremes. The baseline model is affected by two factors: a) higher delays introduce more false negatives in training label, making the target variable increasingly sparse, b)

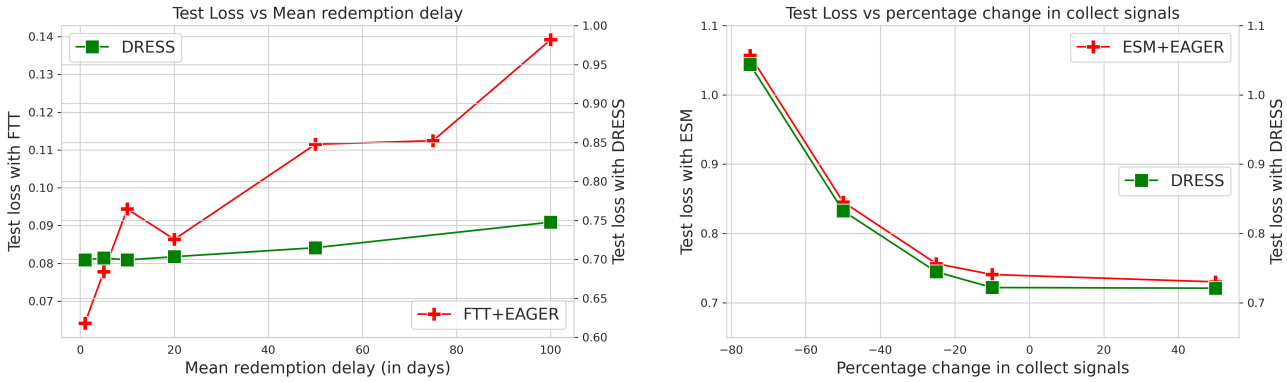

Figure 6: Results of semi-synthetic experiments on the Coupons dataset.

redemption delay introduces drift between train and test labels and the baseline model is evaluated on a dataset highly different from what it has been trained on. Note that the findings from this experiment corroborates the theoretical analysis presented in Section 3.

In the second set of experiments, we vary the mutual information between collects and redemptions and observe its impact on the model performance. We achieve this by randomly dropping and adding a percentage of collect signals. More specifically, for a parameter $\theta \in [-70, 50]$, we drop or add θ percentage of collects i.e. if $\theta = -50$ (say), we drop 50% of existing collect signals. For each value of θ, we report the test loss of Dress and Esm + Eager averaged over 5 runs. In Figure 6 (right), we observe that both models are sensitive to the density of collect signals in the dataset. Dropping collects has an adverse effect on the model performance as it amplifies the sparsity of the target variable.

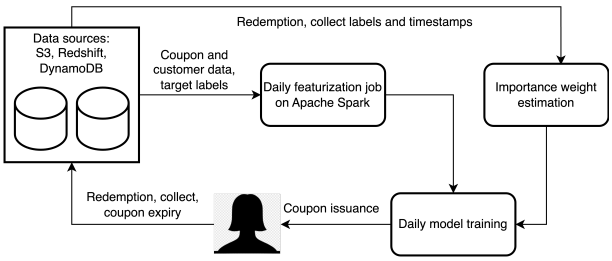

Figure 7: High-level system architecture of Dress.

4.4.3 Online Experiments for Redemption Prediction. To test the efficacy of the framework, we conduct an A/B test where we issue coupons to customers based on the estimated likelihood of redemption. For the treatment group, we de-bias the model training using Dress whereas, the control group uses the Eager strategy for delay correction. The baseline model is kept same for both the cohorts. The experiment was run for a month in an emerging marketplace. A/B test results indicate +10 basis point improvement in number of coupon redemptions. The model has been deployed to production (refer to Figure 7) and is invoked for all coupon issuances.

5 RELATED WORK

Delayed Feedback: There are two schools of thought within the literature on DF. The first stream, founded by the seminal work [3], treats the immature labels as a latent, random variable and employs expectation-maximization (EM) to estimate them in tandem with the conversion prediction. More recently, [19] and [4] carried this forward, with the latter arguing that EM can mitigate the bias inherent in the importance sampling (IS). However, EM, being iterative, puts more demand on the runtime and cost associated with training, and is detrimental especially when training runs daily. On the other hand, the IS line of work minimizes a importance-sampled version of the empirical loss in order to correct for drift introduced by immature labels and duplicated instances in training dataset, an artifact of immutability in a streaming training setup. [25] and the references therein illuminate the underlying design principles behind performing IS in a streaming setup - and thus are not directly applicable to our batch setup.

Entire Space Model: ESM was introduced in the landmark work [12] in the context of conversion modeling in display advertisement, and was further developed by [23]. This line of work exploits the abundance of pre-conversion events, such as click and add-to-wishlist, in order to model the rarer conversion event. [22] combines ESM with DF for display advertisement conversion modeling, and to the best of our knowledge, the only published work to do so. However, [22] employs EM-style iterations , thus rendering Dress to be the first attempt at combining ESM and DF via IS, a cheaper alternative.

6 CONCLUSION AND FUTURE WORK

We consider the problem of issuing coupons to customers with the goal of maximizing number of redemptions. The coupon ranking problem is challenging because of data sparsity and delayed feedback of the target label. We present a novel algorithm Dress which presents a unified architecture to address both these challenges seamlessly. Experimental evidence suggests that the model outperforms traditional baselines for coupon redemption modeling. As a next step, we will experiment with enhancements of Dress and evaluate our algorithm on other recommendation problem scenarios.

REFERENCES

[1] D. K. Agarwal, B.-C. Chen, Q. He, Z. Hua, G. Lebanon, Y. Ma, P. K. Shivaswamy, H.-P. Tseng, J. Yang, and L. Zhang. Personalizing linkedin feed. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015.

[2] M. Beraha, A. M. Metelli, M. Papini, A. Tirinzoni, and M. Restelli. Feature Selection via Mutual Information: New Theoretical Insights. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9, Budapest, Hungary, July 2019. IEEE.

[3] O. Chapelle. Modeling delayed feedback in display advertising. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, page 1097–1105, New York, NY, USA, 2014. Association for Computing Machinery.

[4] Y. Chen, J. Jin, H. Zhao, P. Wang, G. Liu, J. Xu, and B. Zheng. Asymptotically Unbiased Estimation for Delayed Feedback Modeling via Label Correction. WWW, 2022.

[5] W. Feng and J. Wang. Retweet or not? personalized tweet re-ranking. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, page 577–586, New York, NY, USA, 2013. Association for Computing Machinery.

[6] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko. Revisiting deep learning models for tabular data. In *NeurIPS*, 2021.

[7] D. Hillard, S. Schroedl, E. Manavoglu, H. Raghavan, and C. Leggetter. Improving ad relevance in sponsored search. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, page 361–370, New York, NY, USA, 2010. Association for Computing Machinery.

[8] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org, 2015.

[9] P. W. Koh, K.-S. Ang, H. H. K. Teo, and P. Liang. On the Accuracy of Influence Functions for Measuring Group Effects, Nov. 2019.

[10] P. W. Koh and P. Liang. Understanding Black-box Predictions via Influence Functions. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1885–1894. PMLR, July 2017.

[11] S. Kong, Y. Shen, and L. Huang. RESOLVING TRAINING BIASES VIA INFLUENCE- BASED DATA RELABELING. 2022.

[12] X. Ma, L. Zhao, G. Huang, Z. Wang, Z. Hu, X. Zhu, and K. Gai. 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*, SIGIR '18, page 1137–1140, New York, NY, USA, 2018. Association for Computing Machinery.

[13] Y. Ni, D. Ou, S. Liu, X. Li, W. Ou, A. Zeng, and L. Si. Perceive your users in depth: Learning universal user representations from multiple e-commerce tasks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 596–605, New York, NY, USA, 2018. Association for Computing Machinery.

[14] B. Pang, C. Li, Y. Liu, J. Lian, J. Zhao, H. Sun, W. Deng, X. Xie, and Q. Zhang. Improving relevance modeling via heterogeneous behavior graph learning in bing ads. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 3713–3721, 2022.

[15] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS 2017 Workshop on Autodiff*, 2017.

[16] J. Rappaz, J. McAuley, and K. Aberer. Recommendation on live-streaming platforms: Dynamic availability and repeat consumption. In *Proceedings of the 15th ACM Conference on Recommender Systems*, RecSys '21, page 390–399, New York, NY, USA, 2021. Association for Computing Machinery.

[17] D. Sorokina and E. Cantu-Paz. Amazon search: The joy of ranking products. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, page 459–460, 2016.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, jan 2014.

[19] Y. Su, L. Zhang, Q. Dai, B. Zhang, J. Yan, D. Wang, Y. Bao, S. Xu, Y. He, and W. Yan. An Attention-based Model for Conversion Rate Prediction with Delayed Feedback via Post-click Calibration. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 3522–3528, Yokohama, Japan, July 2020. International Joint Conferences on Artificial Intelligence Organization.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[21] J. Wang, P. Huang, H. Zhao, Z. Zhang, B. Zhao, and D. L. Lee. Billion-scale commodity embedding for e-commerce recommendation in alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 839–848, New York, NY, USA, 2018. Association for Computing Machinery.

[22] Y. Wang, J. Zhang, Q. Da, and A. Zeng. Delayed Feedback Modeling for the Entire Space Conversion Rate Prediction, 2021.

[23] H. Wen, J. Zhang, Y. Wang, F. Lv, W. Bao, Q. Lin, and K. Yang. *Entire Space Multi-Task Modeling via Post-Click Behavior Decomposition for Conversion Rate Prediction*, page 2377–2386. Association for Computing Machinery, New York, NY, USA, 2020.

[24] C. Wu, F. Wu, T. Qi, Q. Liu, X. Tian, J. Li, W. He, Y. Huang, and X. Xie. Feedrec: News feed recommendation with various user feedbacks. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 2088–2097, 2022.

[25] J. Yang, X. Li, S. Han, T. Zhuang, D. Zhan, X. Zeng, and B. Tong. Capturing delayed feedback in conversion rate prediction via elapsed-time sampling. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, pages 4582–4589. AAAI Press, 2021.

[26] H. Zhu, D. Chang, Z. Xu, P. Zhang, X. Li, J. He, H. Li, J. Xu, and K. Gai. *Joint Optimization of Tree-Based Index and Deep Model for Recommender Systems*. Curran Associates Inc., Red Hook, NY, USA, 2019.

[27] H. Zhu, X. Li, P. Zhang, G. Li, J. He, H. Li, and K. Gai. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 1079–1088, New York, NY, USA, 2018. Association for Computing Machinery.

APPENDIX

A DERIVATION

Case III ($y^r = 0, y^c = 1$). This scenario encompasses two possibilities: true negative redemption and delayed positive redemption. The observed probability is the net sum of these two outcomes. We can write $q(y^r = 0, y^c = 1 \mid x) = p(y^r = 0, y^c = 1, \delta^c \leq a \mid x) + p(y^r = 1, y^c = 1, \delta^c \leq a, \delta^r > a \mid x)$.

Case IV ($y^r = 0, y^c = 0$). We explore all four possibilities for the observed labels. (a) true negative clip and redemption: in this case, the observed probability is same as $p(y^r = 0, y^c = 0 \mid x)$, (b) delayed redemption: $p(y^r = 1, y^c = 0, \delta^r > a \mid x)$, (c) delayed clip: $p(y^r = 0, y^c = 1, \delta^c > a \mid x)$, (d) delayed clip and redemption: $p(y^c = 1, y^r = 1, \delta^c > a, \delta^r > a \mid x)$. The observed probability is the sum of probabilities for all the four scenarios above,

$$\begin{aligned} q(y^c = 0, y^r = 0 \mid x) = {} & p(y^r = 0, y^c = 0 \mid x) \\ & + p(y^r = 1, y^c = 0, \delta^r > a \mid x) \\ & + p(y^r = 0, y^c = 1, \delta^c > a \mid x) \\ & + p(y^c = 1, y^r = 1, \delta^c > a, \delta^r > a \mid x) \end{aligned} \quad (15)$$

B PROOF OF THEOREM 3.1

Proof. We note that Flip selectively flips some positive labels to negative, $y_i^r \mapsto z_i^r$, where $y_i^r = 1, z_i^r = 0$. Therefore, $\ell(z_i^r, f_r(x_i;\widehat{\theta})) = -\log(1 - f_r(x_i;\widehat{\theta}))$, and, $\ell(y_i^r, f_r(x_i;\widehat{\theta})) = -\log(f_r(x_i;\widehat{\theta}))$.

Therefore, $\mathcal{L}^M(\widehat{\theta^{\mathsf{FL}}}) - \mathcal{L}^M(\widehat{\theta})$ becomes,

$$= \sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \epsilon_i \Phi_i^{\mathsf{FL}} \qquad \text{(From [9])}$$

$$= -\sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \epsilon_i \nabla_\theta \mathcal{L}^M(\widehat{\theta}) H_{\widehat{\theta}}^{-1} \qquad \text{(From Eq.13)}$$
$$\left(\nabla_\theta \ell(z_i^r, f_r(x_i;\widehat{\theta})) - \nabla_\theta \ell(y_i^r, f_r(x_i;\widehat{\theta}))\right)$$

$$= -\sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \epsilon_i \nabla_\theta \mathcal{L}^M(\widehat{\theta}) H_{\widehat{\theta}}^{-1}$$
$$\left(-\frac{\nabla_\theta f_r(x_i;\widehat{\theta})}{1 - f_r(x_i;\widehat{\theta})} - \frac{\nabla_\theta f_r(x_i;\widehat{\theta})}{f_r(x_i;\widehat{\theta})}\right)$$

$$= \sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \epsilon_i \nabla_\theta \mathcal{L}^M(\widehat{\theta}) H_{\widehat{\theta}}^{-1} \frac{\nabla_\theta f_r(x_i;\widehat{\theta})}{f_r(x_i;\widehat{\theta})}$$
$$\left(\frac{f_r(x_i;\widehat{\theta})}{1 - f_r(x_i;\widehat{\theta})} + 1\right)$$

$$= -\sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \epsilon_i \left(-\nabla_\theta \mathcal{L}^M(\widehat{\theta}) H_{\widehat{\theta}}^{-1} \nabla_\theta \ell(y_i^r, f_r(x_i;\widehat{\theta}))\right)$$
$$\left(\frac{1}{1 - f_r(x_i;\widehat{\theta})}\right)$$

$$= -\sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \epsilon_i \frac{\Phi_i^{\mathsf{IS}}}{1 - f_r(x_i;\widehat{\theta})} \qquad \text{(From Eq.14)}$$

$$= -\frac{1}{N} \sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \frac{\Phi_i^{\mathsf{IS}}}{1 - f_r(x_i;\widehat{\theta})} \qquad \text{(From §3.1)} \quad \square$$

C PROOF OF THEOREM 3.2

Proof. Since Theorem 3.2 assumes a perfect true-negative classifier, $g_{tn}^r(x_i, a_i)$, following Eq. 4, Importance Sample effectively discards all false-negatives in $\mathcal{D}_t^{\mathsf{FN}}$ from the training data, but leaves the other negative instances intact.

Similarly, all positive instances are up-weighted with an importance weight suggested by Eq. 5. Moreover, as noted in §2.1.1, $p(\delta_i^r \le a_i | x_i, y_i^r = 1) \ge \beta$, where β is a percentile of wait time distribution.

Thus, $\mathcal{L}^M(\widehat{\theta^{\mathsf{IS}\circ\mathsf{FL}}}) - \mathcal{L}^M(\widehat{\theta^{\mathsf{FL}}})$ becomes,

$$= \sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \epsilon_i \Phi_i^{\mathsf{IS}} + \sum_{i\in\mathcal{D}_t | y_i^r=1} \epsilon_i \Phi_i^{\mathsf{IS}} \qquad \text{(From [9])}$$

$$= -\frac{1}{N} \sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \Phi_i^{\mathsf{IS}} + \sum_{i\in\mathcal{D}_t | y_i^r=1} \epsilon_i \Phi_i^{\mathsf{IS}} \qquad \text{(From Eq. 4 \& §3.1)}$$

$$= -\frac{1}{N} \sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \Phi_i^{\mathsf{IS}} + \qquad \text{(From Eq. 5 \& §3.1)}$$
$$\frac{1}{N} \sum_{i\in\mathcal{D}_t | y_i^r=1} \left(\frac{1}{p(\delta_i^r \le a_i | x_i, y_i^r = 1)} - 1\right) \Phi_i^{\mathsf{IS}}$$

$$\le -\frac{1}{N} \sum_{i\in\mathcal{D}_t^{\mathsf{FN}}} \Phi_i^{\mathsf{IS}} + \frac{1}{N} \sum_{i\in\mathcal{D}_t | y_i^r=1} \left(\frac{1}{\beta} - 1\right) \Phi_i^{\mathsf{IS}} \qquad \text{(From §2.1.1)}$$

$$= -\frac{1}{N} \sum_{i\in\mathcal{D}_t^{\mathsf{IS}}} \Phi_i^{\mathsf{IS}} + \frac{1}{N} \sum_{i\in\mathcal{D}_t | y_i^r=1} \frac{1-\beta}{\beta} \Phi_i^{\mathsf{IS}} \qquad \square$$