BEYOND ONE EPOCH: UNCERTAINTY-WEIGHTED SENSITIVITY REGULARIZATION FOR RECOMMENDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recommendation models with sparse embeddings and a shared consumer often exhibit the *one-epoch phenomenon*: a second epoch lowers training loss while sharply degrading generalization. We present a view based on the violation of the prequential principle. On the first epoch, an example's label has not affected the embedding rows used to score it. On later epochs, those rows contain a displacement induced by the labels earlier update. This creates an incentive for the shared consumer to exploit this displacement in subsequent epochs, which fails to generalize. We call this *self-influence asymmetry*. Using an exact scalar model and local influence analysis, we connect this mismatch to the uncertainty in the embeddings and the consumers incentive to exploit it in subsequent epochs. We verify this hypothesis using an embedding-consumer-update interventions in deep recommendation models and propose uncertainty-weighted sensitivity regularization (UWSR) which counteracts this mismatch by augmenting the loss function to penalize the consumer for relying on uncertain embeddings. Unlike existing remedies, UWSR preserves the learned embeddings and across three benchmarks, four-epoch UWSR reduces test cross-entropy by 1.38%–6.78% and improves AUC by 0.0058–0.0231 relative to one-epoch training.

1 INTRODUCTION

Recommendation models score the likelihood of a user interacting with an item, such as predicting click-through rates for displayed content. Under the hood, these systems map high-dimensional categorical inputs, like user and item IDs, to learned vector representations stored in embedding tables. As updates apply only to entities active in a given batch of examples during training, these embedding lookups are inherently sparse. A shared consumer, usually a dense neural network, then combines these retrieved vectors with other features to compute the final prediction. As modern systems scale to encompass billions of unique entities, these sparse embedding tables dominate the model parameter count compared to the consumer by several orders of magnitude.

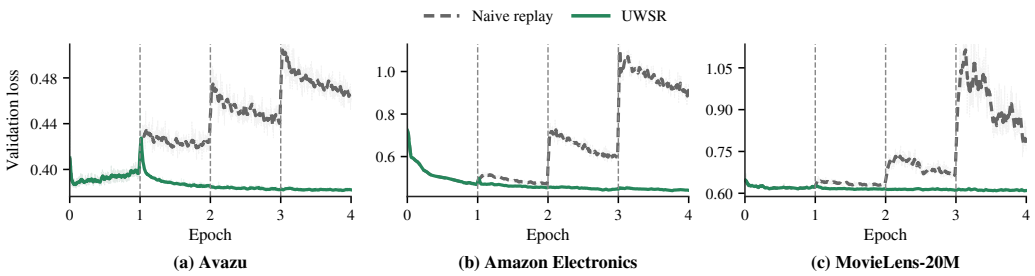


Figure 1: Validation Loss on Avazu, Amazon Electronics, and MovieLens-20M datasets comparing naive multi-epoch training with proposed UWSR method. Naive training displays one-epoch phenomenon of increasing validation loss at epoch boundary, remediated via UWSR.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

While dense neural networks may overfit when trained for multiple epochs, recommendation models often exhibit a peculiar failure at the start of the second epoch: out of sample performance deteriorates abruptly while training loss improves. Zhang et al. (2022) termed this failure the *one-epoch phenomenon*, hypothesizing that first-seen and revisited training examples induce fundamentally different joint distributions of embeddings and labels. They attribute this discrepancy to sparse features and the rapid adaptation of the shared consumer. Figure 1 demonstrates this phenomenon on three benchmarks. To address this, mitigations typically intervene directly on the sparse parameters. Common strategies include resetting embedding tables between epochs (Fan et al., 2024) or attenuating row magnitudes according to their occurrence frequencies (Li & Lyu, 2026).

In this paper, we analyze the one-epoch phenomenon through the lens of the *prequential principle*. During the first epoch, training is *prequential*: the forward pass scores each example before its label can influence the model parameters. During subsequent epochs, or replay, the model consumes embedding features whose state is already conditioned on the very target being predicted. This causes an implicit augmentation in the loss function at the epoch boundary where the consumer relies on the stored information in the embeddings during replay. We call the resulting train-time change in the effective objective and optima *self-influence asymmetry*. We quantify this implicit change in the objective function and optima for an exact solvable scalar model and a local neural approximation, showing the exacerbation of the effect with the reduction in the precision of the embeddings, explaining previous empirical observations of the one epoch phenomenon being more severe in datasets whose embeddings have low support .

Pathologies caused by breaking prequential training have long been recognized. The generated-regressor problem (Pagan, 1984) studies downstream regressions whose covariates are themselves estimated. For example, suppose a student's exam score y is predicted from their mean test score x, but that mean includes the same exam score y. The fitted relationship will appear stronger than it is for a new exam excluded from the construction of x. In the limiting case in which the mean contains only that exam, $x = y$, Pagan (1984) further shows that when this reuse cannot be avoided, the estimation uncertainty in the generated regressor must itself be propagated through the downstream fit, or inference on it will be overconfident. These results point to two complementary remediations: exclude an outcome from the fitted inputs used to predict it, or, when exclusion is impractical, account for the uncertainty in those inputs. Modern supervised learning favors exclusion, using techniques like out-of-fold target encoding (Micci-Barreca, 2001), ordered encodings (Prokhorenkova et al., 2018), and cross-fitting (Chernozhukov et al., 2018) to ensure no observation constructs its own features. However, because deep recommendation models jointly optimize embeddings and downstream consumers, multi-epoch training is antithetical to this principle.

Our exact scalar model and local neural analysis show that the resulting self-influence asymmetry is governed by the prediction's sensitivity to each fitted embedding rows, and weighted by their uncertainty. This identifies a tractable quantity to control without interrupting end-to-end embedding training. With these insights, we propose uncertainty-weighted sensitivity regularization (UWSR). a method that penalizes the downstream network's sensitivity to poorly determined sparse rows, making it explicitly costly to depend on them. UWSR computes this gradient penalty via standard double backpropagation (Drucker & Le Cun, 1992), acting as a functional regularizer on the consumer instead of the embeddings. Thus, unlike direct sparse-parameter interventions, it counteracts the network's incentive to exploit the previous influence of the label on the embeddings, while leaving them free to be optimized otherwise.

Contributions. In summary, our key contributions are:

1. We show that *self-influence asymmetry*, an effect echoing the classical generated-regressor problem, contributes to the one-epoch phenomenon. Using an exactly solvable scalar model and local rowwise analysis on deep recommendation models reveals a mismatch between the consumer a training example rewards and the consumer a fresh example rewards. We use an embedding-consumer-update-order intervention to verify this hypothesis.
2. We derive UWSR from *self-influence asymmetry*, which counteracts the mismatch and provide a computationally tractable method; and
3. We empirically show that UWSR enables multi-epoch training across three datasets (MovieLens-20M, Amazon Electronics, and Avazu), consistently outperforming both single-epoch checkpoints and multi-epoch baselines from state-of-the-art methods.

2 Related Work

The one-epoch phenomenon. The sharp degradation of deep recommendation models during the second epoch Zhang et al. (2022) has previously been attributed to distributional shifts in sparse embeddings and rapid network adaptation. However, this perspective conflates general embedding maturity with the leakage of the currently scored label. Prior experiments find that standard regularizers such as ℓ_2 decay and embedding dropout do not remove the failure (Zhang et al., 2022); proposed remedies instead reset embedding tables (Fan et al., 2024) or attenuate their magnitudes (Li & Lyu, 2026).

Target exclusion and fitted representations. Established statistical frameworks recognize that target encoding leaks information if a representation includes the instance's own label (Micci-Barreca, 2001). To prevent this, they explicitly enforce target exclusion using techniques such as ordered statistics (Prokhorenkova et al., 2018) or cross-fitting (Chernozhukov et al., 2018). Generated-regressor theory studies the downstream consequences of plugging in an estimated quantity (Pagan, 1984). Although these fields motivate target exclusion and uncertainty propagation, they do not address the deep-learning setting in which intermediate representations (sparse embeddings) and their downstream consumers are jointly and repeatedly optimized.

Prequential prediction, influence, and leverage. Our theoretical framework builds on the prequential principle, which dictates that a model must generate a prediction before observing its target (Dawid, 1984). To quantify the exact mathematical consequences when this temporal separation breaks down, we draw upon influence functions, which decompose a fitted parameter into leave-one-out and self-influence components (Koh & Liang, 2017). This self-influence is closely tied to statistical leverage and its extension to classification via generalized leverage (Pregibon, 1981) - which formalizes how strongly an individual observation shifts its own fitted value. Efron's covariance-penalty theory expresses in-sample optimism through the covariance between a response and its fitted prediction (Efron, 2004).

Input sensitivity and information-weighted parameter methods. The theoretical equivalence between additive input noise and derivative penalties is a classical result (Bishop, 1995), laying the groundwork for techniques such as double backpropagation and input-Jacobian regularization, which explicitly penalize a model's sensitivity to external inputs (Drucker & Le Cun, 1992). A parallel literature addresses parameter uncertainty: methods such as AdaGrad, K-FAC, and elastic weight consolidation use expected curvature or Fisher information to selectively scale parameter updates and penalize displacement (Duchi et al., 2011; Martens & Grosse, 2015; Kirkpatrick et al., 2017), while Bayesian neural networks achieve a similar goal through uncertainty sampling (Blundell et al., 2015). Our proposed UWSR bridges these approaches, applying input-sensitivity penalties not to external data, but to the model's own fitted, information-weighted representations.

3 Self-Influence asymmetry

In this section, we demonstrate the mechanics of self-influence asymmetry and its dependence on the support of an embedding row. We first use an exactly solvable scalar model to exactly quantify this asymmetry and then generalize the concept to neural networks. Finally, we demonstrate the impact of this phenomenon using an embedding-consumer-update-order intervention experiment that deliberately breaks prequential training within the first epoch.

3.1 Scalar Model of the Leave-In/Leave-Out Asymmetry

In this section, we use a scalar model to quantify self-influence asymmetry and how replay implicitly distorts the prequential training objective function, exacerbated as the support for an embedding reduces. We then propose a method to resolve this asymmetry by augmenting the loss function.

Consider an ordered dataset of interactions $\mathcal{D} = \{(x_t, y_t)\}_{t=1}^{T}$, where x_t maps to a single scalar embedding e_x in the embedding table E and y_t is the continuous target estimated as $\hat{y}_t = we_{x_t}$, where w represents the shared consumer. For an entity x that occurs N_x times in $\mathcal{D}$, assume $y_k = \mu_x + \epsilon_k$ for $k \in \{1, \ldots, N_x\}$, where μ_x denotes the latent signal associated with x, and ϵ_k denotes the noise

specific to occurrence k, independent of μ_x. Assume both are zero-mean, with $\mathrm{Var}(\mu_x) = \tau^2$ and $\mathrm{Var}(\epsilon_k) = \nu^2$ and the noise instances $\{\epsilon_k\}$ are i.i.d.

To quantify self-influence asymmetry, we use a two-stage setup: we first fit embedding e_x on observations of x in $\mathcal{D}$, then hold it fixed to derive the optimal consumer w across in-sample and fresh targets. We fit the embedding e_x using the empirical squared loss for entity x:

$$L_{\mathrm{fit},x}(e) = \frac{1}{2N_x}\sum_{k=1}^{N_x}(y_k - e)^2, \qquad \hat{e}_x = \arg\min_{e\in\mathbb{R}} L_{\mathrm{fit},x}(e) = \frac{1}{N_x}\sum_{k=1}^{N_x} y_k = \mu_x + \bar{\epsilon}_x,$$

where $\bar{\epsilon}_x = N_x^{-1}\sum_{k\in\mathcal{K}_x}\epsilon_k$. As ϵ_k is independent of μ_x, $\mathbb{E}[\hat{e}_x \mid \mu_x] = \mu_x$ making $\hat{e}_x$ a noisy estimate of mu_x, with estimation error $\hat{e}_x - \mu_x = \bar{\epsilon}_x$ and variance $\Sigma_x = \mathrm{Var}(\hat{e}_x - \mu_x) = \mathrm{Var}(\bar{\epsilon}_x) = \frac{\nu^2}{N_x}$.

Having characterized the fitted embedding, we now turn to study how strongly the consumer should rely on it, where $w = 0$ ignores the embedding and $w = 1$ uses it without attenuation. Holding $\hat{e}_x$ fixed, we evaluate the reliance of consumer w by comparing two targets scored using it: a fresh target $y_{\text{fresh}} = \mu_x + \epsilon_{\text{fresh}}$ (ϵ_{fresh} is i.i.d. with $\{\epsilon_k\}$) and a replayed target $y_{\text{replay}} = \mu_x + \epsilon_{\text{replay}}$ (chosen form $\mathcal{D}$, used to fit $\hat{e}_x$). We fit w using the loss functions:

$$L_{\mathrm{fresh},x}(w) = \mathbb{E}\left[(y_{\mathrm{new}} - w\hat{e}_x)^2\right], \qquad L_{\mathrm{replay},x}(w) = \mathbb{E}\left[(y_{\text{replay}} - w\hat{e}_x)^2\right].$$

The two losses differ solely through the target and fitted embedding covariance. For fresh targets, ϵ_{new} is independent of $\hat{e}_x$, thus $\mathrm{Cov}(\hat{e}_x, y_{\mathrm{new}}) = \mathrm{Cov}\left(\mu_x + \bar{\epsilon}_x, \mu_x + \epsilon_{\mathrm{new}}\right) = \mathrm{Var}(\mu_x) = \tau^2$ and,

$$L_{\mathrm{fresh},x}(w) = (1-w)^2\tau^2 + \nu^2 + w^2\Sigma_x.$$

For replay, y_{replay} contributed to $\hat{e}_x$, hence $\mathrm{Cov}(\hat{e}_x, y_{\text{replay}}) = \tau^2 + \mathrm{Cov}(\bar{\epsilon}_x, \epsilon_{\text{replay}}) = \tau^2 + \Sigma_x$ and,

$$L_{\mathrm{replay},x}(w) = L_{\mathrm{fresh},x}(w) - 2w\Sigma_x.$$

Thus, for $w \geq 0$, replay obtains an additional loss reduction of $2w\Sigma_x$ that is absent during fresh evaluation. Solving first-order optimality conditions for w for the above two loss functions yields,

$$\boxed{w^\star_{\mathrm{fresh},x} = \frac{\tau^2}{\tau^2 + \Sigma_x}, \qquad w^\star_{\mathrm{replay},x} = 1.}$$

Thus, fresh evaluation favors shrinking the contribution of an uncertain embedding, while replay favors using the embedding without attenuation. If the consumer adopts the replay-optimal value $w^\star_{\mathrm{replay},x} = 1$, its excess loss on a fresh target is

$$L_{\mathrm{fresh},x}(w^\star_{\mathrm{replay},x}) - L_{\mathrm{fresh},x}(w^\star_{\mathrm{fresh},x}) = \frac{\Sigma_x^2}{\tau^2 + \Sigma_x}.$$

Since $\Sigma_x = \nu^2/N_x$, both the difference between the two optimal consumers and the resulting excess fresh loss grow with decrease in the embedding support N_x.

3.2 Generalizing to Neural Networks

We now demonstrate self-influence asymmetry in neural recommendation systems. Consider an ordered dataset $\mathcal{D} = \{(x_t, y_t)\}_{t=1}^T$ with targets $y_t \in \{0, 1\}$. The model estimates $\hat{y}_t$, the probability $y_t = 1$, as $\hat{y}_t = \sigma(z_t) = (1 + \exp(-z_t))^{-1}$, governed by two parameterized components:

- **Sparse Embedding Tables** $\{E^{(F)}\}_{F=1}^M$: A collection of M embedding tables, with table $E^{(F)} \in \mathbb{R}^{n_F \times d_F}$ containing n_F rows of dimension d_F. For an input x_t, $\mathcal{A}(x_t)$ denotes the set of active table-row pairs and $E_{x_t} := \{e_{F,i} : (F, i) \in \mathcal{A}(x_t)\}$ the retrieved embeddings.
- **Shared Consumer Network (w)**: Combines E_{x_t} and remaining continuous features to output scalar logit $z_t = z(E_{x_t}, \mathbf{w})$.

The model uses the binary cross-entropy (BCE) loss: $\ell(z_t, y_t) = -y_t \log \hat{y}_t - (1 - y_t)\log(1 - \hat{y}_t)$. Following the two-stage abstraction for the scalar model, we first isolate the fitting of an embedding from the downstream consumer using a local row-wise analysis. For a fixed occurrence t and active

pair $(F, i) \in \mathcal{A}(x_t)$, we hold the remaining embeddings and consumer fixed. Let $\mathcal{G}_{t^-}$ denote the pre-update information set containing x_t and prior interactions, excluding y_t and updates from (x_t, y_t). The pre-update logit z_t and prediction $\hat{y}_t$ are deterministic given $\mathcal{G}_{t^-}$. We define the conditionally optimal prediction $\hat{y}_t^\star = \Pr(y_t = 1 \mid \mathcal{G}_{t^-})$, its logit $z_t^\star = \sigma^{-1}(\hat{y}_t^\star)$, and logit error $\Delta z_t = z_t - z_t^\star$. To evaluate prediction costs, we introduce three core quantities to characterize the active embedding row (F, i) and the network's reliance on it:

- **Fitted Row & Error:** Let $\hat{e}_{F,i}^-$ denote the fitted value of the active row in $\mathcal{G}_{t^-}$, and let $e_{F,i}^*$ denote the reference value yielding target logit $z_t^\star$. Thus, the pre-update row error is defined as $\Delta e_{F,i}^- = \hat{e}_{F,i}^- - e_{F,i}^*$.
- **Embedding Uncertainty:** We model row parameter uncertainty due to finite observations via a Laplace approximation with pre-update precision $\mathcal{I}_{F,i}^- > 0$ (the damped row-wise generalized Gauss-Newton information[1]). We denote the expectation over this posterior as $\mathbb{E}_*$, thus the error covariance satisfies $\mathbb{E}_*[\Delta e_{F,i}^- (\Delta e_{F,i}^-)^\top] = (\mathcal{I}_{F,i}^-)^{-1}$.
- **Local Logit Sensitivity:** Extending beyond linear models where sensitivity is constant, we capture the neural consumer's local reliance on $e_{F,i}$ via the logit sensitivity vector $s_{F,i,t} = \nabla_{e_{F,i}} z(E_{x_t}, \mathbf{w})|_{e_{F,i}=e_{F,i}^*}$. A first-order Taylor expansion yields $\Delta z_t \approx s_{F,i,t}^\top \Delta e_{F,i}^-$.

With this setup, we first characterize the cost of this embedding uncertainty during a fresh evaluation, and then show how replaying this occurrence exploits this embedding uncertainity.

3.2.1 Implicit Penalization of Uncertain Embeddings in Prequential Training

This Section demonstrates how prequential training naturally penalizes reliance on uncertain embedding rows, by implicitly introducing an excess loss penalty proportional to embedding uncertainty, thus showing that first-epoch training naturally penalizes the downstream network for relying on poorly determined representations. For a fixed example t, consider the Taylor expansion of $l(z_t, y_t)$ around z_t^*:

$$\ell(z_t, y_t) - \ell(z_t^*, y_t) = (\hat{y}_t^* - y_t)\Delta z_t + \frac{1}{2}\hat{y}_t^*(1 - \hat{y}_t^*)(\Delta z_t)^2 + \mathcal{O}(|\Delta z_t|^3) \tag{1}$$

As Δz_t is deterministic given $\mathcal{G}_{t^-}$, and y_t has conditional expectation $\mathbb{E}[y_t \mid \mathcal{G}_{t^-}] = \hat{y}_t^*$, the first-order interaction in (1) gives $\mathbb{E}[(\hat{y}_t^* - y_t)\Delta z_t \mid \mathcal{G}_{t-}] = \Delta z_t(\hat{y}_t^* - \mathbb{E}[y_t \mid \mathcal{G}_{t-}]) = 0$. Taking the conditional expectation of (1) while removing higher order terms yields,

$$\mathbb{E}[\ell(z_t, y_t) - \ell(z_t^*, y_t) \mid \mathcal{G}_{t-}] \approx \frac{1}{2}\hat{y}_t^*(1 - \hat{y}_t^*)(\Delta z_t)^2 \geq 0 \tag{2}$$

Substituting $\Delta z_t \approx s_{F,i,t}^\top \Delta e_{F,i}^-$ and taking expectation $\mathbb{E}_*$ over the posterior row uncertainty yields:

$$\mathbb{E}_*[(\Delta z_t)^2] \approx s_{F,i,t}^\top \mathbb{E}_*[\Delta e_{F,i}^- (\Delta e_{F,i}^-)^\top] s_{F,i,t} = s_{F,i,t}^\top (\mathcal{I}_{F,i}^-)^{-1} s_{F,i,t} \tag{3}$$

Combining (2) and (3) gives the expected local excess risk under first-epoch prequential training:

$$\boxed{\mathbb{E}_\star\left[\mathbb{E}[\ell(z_t, y_t) - \ell(z_t^\star, y_t) \,|\, \mathcal{G}_{t^-}]\right] \approx \frac{1}{2}\hat{y}_t^\star(1 - \hat{y}_t^\star) s_{F,i,t}^\top (\mathcal{I}_{F,i}^-)^{-1} s_{F,i,t} \geq 0,} \tag{4}$$

This local excess demonstrates that prequential training naturally exposes a positive quadratic penalty on downstream logit sensitivity. Predictions incur the highest excess loss when the shared consumer network exhibits high sensitivity $s_{F,i,t}$ toward poorly informed embedding directions, that rows where $\mathcal{I}_{F,i}^-$ is small.

3.2.2 Replay-Induced Exploitation of Uncertain Embeddings

In this subsection, we show how replay promotes the consumer to rely on uncertain embeddings and cause the self-influence asymmetry. Let $r_t = y_t - \hat{y}_t$ denote the prediction residual, where $\hat{y}_t$ is

[1]The row-wise GGN forms the curvature block for $e_{F,i}$, scaling logit gradient outer products by BCE curvature. Damping guarantees invertibility, making $(\mathcal{I}_{F,i}^-)^{-1}$ a valid posterior covariance for rarely observed rows.

deterministic given $\mathcal{G}_{t^-}$. When occurrence t updates the model, it writes a label-aligned component into the active row $e_{F,i}$. We define this update as the self-induced row displacement, denoted by $\delta e^{\text{self}}_{F,i,t}$. When occurrence t is subsequently replayed, the shared consumer scores this modified embedding row, shifting the output logit. We define this output shift as the logit displacement, u_t.

To measure the change in loss post replay, let y'_t denote an independent target label for x_t which did not contribute to updating the embedding rows. Comparing the loss on the observed label y_t against the expected loss on an independent target y'_t defines the leave-in optimism as,

$$D_{F,i,t} = \ell(z_t + u_t, y_t) - \ell(z_t, y_t) - \mathbb{E}[\ell(z_t + u_t, y'_t) - \ell(z_t, y'_t) \mid \mathcal{G}_{t^-}, x_t, y_t].$$

We formalize the relationship between these quantities in the proposition to follow.

Proposition 1. *In the local Generalized Gauss-Newton (GGN) row model with post-update row information $\mathcal{I}_{F,i} = \mathcal{I}^-_{F,i} + h_t s_{F,i,t} s^\top_{F,i,t}$ and variance scale $h_t = \hat{y}_t(1-\hat{y}_t)$ evaluated at the model's own pre-update prediction, the following properties hold:*

1. *The self-induced row displacement $\delta e^{\text{self}}_{F,i,t}$ and resulting logit shift u_t satisfy*
$$\delta e^{\text{self}}_{F,i,t} = r_t \mathcal{I}^{-1}_{F,i} s_{F,i,t} \quad \text{and} \quad u_t = r_t \|s_{F,i,t}\|^2_{\mathcal{I}^{-1}_{F,i}} \tag{5}$$

2. *For an independent target $y'_t \sim \text{Bernoulli}(\hat{y}_t)$, the leave-in optimism $D_{F,i,t}$ and its conditional expectation satisfy*
$$D_{F,i,t} = -r_t^2 \|s_{F,i,t}\|^2_{\mathcal{I}^{-1}_{F,i}} \quad \text{and} \quad -\mathbb{E}[D_{F,i,t} \mid \mathcal{G}_{t-}] = h_t \|s_{F,i,t}\|^2_{\mathcal{I}^{-1}_{F,i}} \in [0, 1). \tag{6}$$

Appendix A proves Proposition 1. Comparing (4) and (6) reveals the opposing incentives of prequential training and replay. Target exclusion penalizes logit sensitivity under $\mathcal{I}^{-1}_{F,i,-t}$, whereas self-inclusion rewards it under $\mathcal{I}^{-1}_{F,i}$. As rare or low-support embedding rows accumulate minimal information $\mathcal{I}_{F,i}$, their inverse norm $\|\cdot\|^2_{\mathcal{I}^{-1}_{F,i}}$ is largest, yielding the highest generalized leverage during replay. As multi-epoch training actively breaks prequential training, it actively incentivizes the shared consumer network to inflate its score sensitivity $s_{F,i,t}$ toward poorly determined representations, exploiting target leakage to suppress training loss at the expense of generalization. UWSR is designed to restore the missing positive pressure during later epochs.

3.3 Embedding-Consumer-Update-Order Intervention Experiment

In this section, we present empirical evidence that breaking same-label self-inclusion drives sensitivity inflation by inducing these conditions in the first epoch itself. In ordinary training, both embeddings and consumer are updated within a single forward-backward pass. We alter this process by performing two passes for each minibatch while matching the total updates of standard joint training. In embedding→consumer, only embeddings are updated in the first pass and the examples are rescored to update the consumer. In consumer→embedding, this order is reversed[2].

Intuitively, embedding-first writes a small, label-aligned trace into each active row before the consumer is trained. Rescoring then makes those examples easier to predict, encouraging the consumer to become more sensitive to the modified rows. This lowers training loss but provides no advantage on fresh examples. Consumer-first prevents the consumer from exploiting this trace, so we expect much smaller increases in sensitivity and validation loss. Table 1 reports results after a single epoch. While consumer-first remains close to ordinary training, embedding-first creates a large train loss advantage but substantially increases score sensitivity and worsens validation performance. Figure 2 further shows this inflation is strongest for poorly supported rows. Together, these results demonstrate that same-label self-inclusion is sufficient to induce sensitivity inflation within the first epoch.

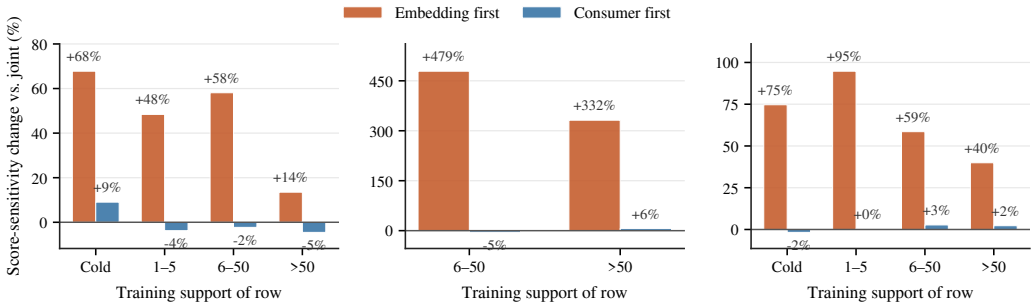
4 Uncertainty-Weighted Sensitivity Regularization

In this section, we propose uncertainty-weighted sensitivity regularization (UWSR), which counteracts self-influence asymmetry by adding a functional regularizer on the dense consumer network, restoring the positive inverse-information penalty present during the first epoch.

[2]The experimental setup employed here is representative of modern recommendation systems, detailed in Section 5

Table 1: One-epoch embedding-consumer-update-order intervention, averaged over two seeds. $\Delta\text{BCE}_{\text{train}}$ is the reduction in same-minibatch training loss after the first block update. $\Delta\text{BCE}_{\text{val}}$ is the validation-loss reduction relative to ordinary training; positive values indicate improvement. ΔS is the relative change in primary-row logit sensitivity relative to joint training.

	Embedding→Consumer			Consumer→Embedding		
Dataset	$\Delta\text{BCE}_{\text{train}}$	$\Delta\text{BCE}_{\text{val}}$	ΔS	$\Delta\text{BCE}_{\text{train}}$	$\Delta\text{BCE}_{\text{val}}$	ΔS
MovieLens-20M	$+1.43\%$	-2.90%	$+54.0\%$	$+0.19\%$	-0.21%	-0.1%
Amazon Electronics	$+10.93\%$	-13.14%	$+474.3\%$	$+0.22\%$	-0.44%	-4.8%
Avazu	$+1.73\%$	-1.86%	$+56.4\%$	$+0.15\%$	$+0.31\%$	$+4.6\%$



(a) Avazu **(b) Amazon Electronics** **(c) MovieLens-20M**

Figure 2: Mean percentage change in primary-row logit sensitivity for embedding-consumer-update-order interventions compared to ordinary joint training for Avazu, Amazon Electronics and MovieLens-20M datasets grouped by row support.

The local self-inclusion advantage is identified as $-r_t^2 s_{F,i,t}^{\mathsf{T}} \mathcal{I}_{F,i}^{-1} s_{F,i,t}$ in (6). As $|r_t| \leq 1$, its magnitude is bounded by $s_{F,i,t}^{\mathsf{T}} \mathcal{I}_{F,i}^{-1} s_{F,i,t}$. With $s_{F,i}(x) = \nabla_{e_{F,i}} z(x)$, UWSR restores this positive pressure by adding the following penalty to the loss after the first epoch:

$$R(x) = \sum_{(F,i)\in\mathcal{A}(x)} s_{F,i}(x)^{\mathsf{T}} \widehat{\Sigma}_{F,i} s_{F,i}(x), \qquad \mathcal{L}_{\text{UWSR}}(x,y) = \ell(z(x),y) + \lambda R(x), \tag{7}$$

where $\widehat{\Sigma}_{F,i}$ is the frozen inverse-information weights. This is the same inverse-information geometry maintained by prequential training, as identified in (4). To deploy UWSR in practice, we estimate $\widehat{\Sigma}_{F,i}$ during epoch one, keep it fixed and apply the sensitivity penalty efficiently subsequent epochs.

- **Estimation of $\widehat{\Sigma}_{F,i}$.** We estimate $\widehat{\Sigma}_{F,i}$ using the the squared-gradient accumulator (Li et al., 2025) that adaptive optimizers such as Adam already maintain, repurposed as an approximation of the diagonal Fisher information. Rather than one scalar per parameter, we accumulate one scalar per embedding row, the same row-wise reduction used by row-wise Adagrad and row-wise Adam; Appendix B.1 gives the exact construction and its per-coordinate ablation. This is not identical to the Fisher (Li et al. (2025) characterize the exact differences between the two), but across five applications it performs comparably to the Fisher while requiring no additional computation, since the accumulator is already computed as part of ordinary optimizer state. We adopt it as our default estimator for this reason; the expected-Fisher and uniform-weight variants are compared directly in Appendix B.3.
- **Computing $s_{F,i}(x)$**: This can be computed efficiently via double backpropagation (Drucker & Le Cun, 1992) and second-order automatic differentiation, available in PyTorch via `create_graph=True`.

5 Results

Evaluation Protocol We evaluate on MovieLens-20M (Harper & Konstan, 2015), AmazonElectronics (McAuley et al., 2015; Zhou et al., 2018), and Avazu (Wang & Cukierski, 2014) using

Table 2: Test-set performance after each epoch, averaged over five seeds. Each endpoint uses the coefficient selected on validation at that endpoint. The right block gives the paired relative BCE reduction against the same seed's one-epoch checkpoint; positive values are better.

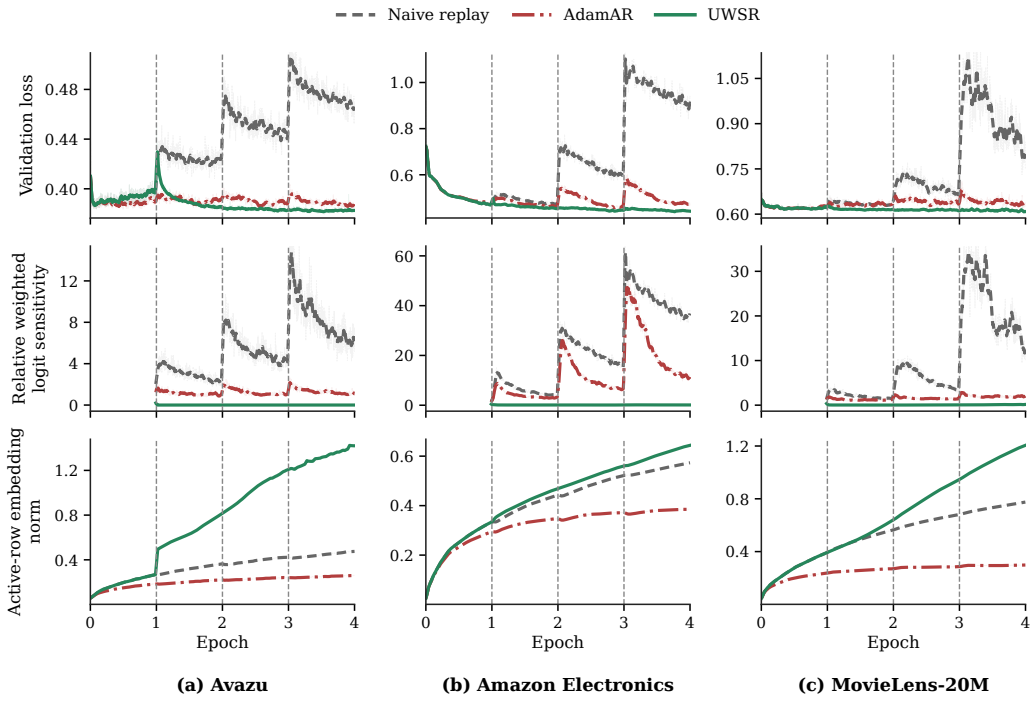
		Test AUC				ΔBCE (%)		
Dataset	Method	E1	E2	E3	E4	E2	E3	E4
Amazon	Naive Replay	0.8523	0.8594	0.8403	0.8233	+0.02	−28.51	−95.43
	MEDA	0.8523	0.8608	0.8644	0.8649	+2.54	+3.62	+3.92
	Freeze until final epoch	0.8523	0.8469	0.8483	0.8460	−1.61	−1.10	−1.74
	Active-row AdamW	0.8523	0.8594	0.7852	0.7811	+0.02	−20.06	−23.54
	Embedding dropout	0.8523	0.8575	0.8453	0.8417	+0.72	−4.03	−7.16
	AdamAR	0.8523	0.8640	0.8660	0.8656	+3.40	+3.07	+2.92
	UWSR Uniform	0.8523	0.8591	0.8556	0.8516	+1.79	−0.33	−1.21
	UWSR	0.8523	**0.8687**	**0.8735**	0.8754	+5.10	+6.43	+6.78
	UWSR Expected Fisher	0.8523	**0.8687**	0.8734	**0.8756**	+5.13	+6.43	+6.93
MovieLens	Naive Replay	0.6766	0.6436	0.5926	0.5097	−1.89	−7.08	−33.66
	MEDA	0.6766	0.6800	0.6711	0.6707	+0.96	+0.03	+0.21
	Freeze until final epoch	0.6766	0.6703	0.6502	0.6560	+0.13	−1.39	−0.90
	Active-row AdamW	0.6762	0.6840	0.6821	0.6777	+0.62	+0.77	+0.10
	Embedding dropout	0.6766	0.6802	0.6570	0.6360	+0.81	−0.67	−2.63
	AdamAR	0.6766	0.6756	0.6661	0.6752	+0.30	−0.69	−0.86
	UWSR Uniform	0.6766	0.6858	**0.6940**	0.6820	+0.90	+2.00	+1.19
	UWSR	0.6766	**0.6951**	0.6892	**0.6824**	+1.97	+1.84	+1.38
	UWSR Expected Fisher	0.6766	0.6947	0.6896	0.6796	+1.94	+1.75	+1.18
Avazu	Naive Replay	0.7405	0.7148	0.6976	0.6837	−4.51	−9.25	−13.85
	MEDA	0.7405	0.7409	0.7425	0.7413	+0.08	+0.03	−0.11
	Freeze until final epoch	0.7405	0.7388	0.7411	0.7401	−0.46	+0.00	−0.20
	Active-row AdamW	0.7348	0.7304	0.7223	0.7191	−1.82	−3.79	−5.76
	Embedding dropout	0.7405	0.6992	0.6961	0.6947	−4.86	−6.37	−7.77
	AdamAR	**0.7489**	0.7481	0.7497	0.7501	+1.35	+1.52	+1.46
	UWSR Uniform	0.7405	0.7480	0.7470	0.7407	+1.32	+1.25	+0.48
	UWSR	0.7405	0.7554	**0.7561**	0.7547	+2.46	+2.43	+2.29
	UWSR Expected Fisher	0.7405	**0.7561**	0.7537	**0.7559**	+2.56	+2.21	+2.49

training, validation, and test partitions. Validation selects all hyperparameters; a disjoint subset of the validation data is used only for the affine-calibration diagnostic. The test partition is not used for selection. We report raw BCE and AUC, with AUC as the primary metric; Appendix C gives dataset details. The model's shared consumer is a compact WuKong-inspired interaction model Zhang et al. (2024) and methods are trained for four epochs unless noted. Training examples are independently reshuffled on every pass, using the same pass-specific permutation for every arm; the epoch-boundary effect therefore does not require replaying the same minibatch order. Appendix D provides further details.

Comparisons We compare one epoch, naive four-epoch replay, MEDA (Fan et al., 2024), and AdamAR (Li & Lyu, 2026), active-row AdamW, embedding dropout (Srivastava et al., 2014), and embeddings frozen until the final epoch. All methods other than AdamAR and active-row AdamW use Adam (Kingma & Ba, 2014) optimizer. Alongside UWSR, we evaluate UWSR Expected Fisher and UWSR Uniform, described in Appendix B.3. As multi-epoch arms use four times the gradient steps of the one-epoch baseline, the comparison asks whether additional training is useful.

Test-Set Performance Table 2 summarizes the empirical results, showing that UWSR shows consistent improvement across all datasets beyonf one epoch against naive replay, one epoch checkpoint, and state of the art methods. Further results on more architectures are summarized on Appendix G. The method also incurs minimal computational overhead, wall-clock cost in Appendix H.

Training diagnosis Figure 3 shows the impact of using UWSR on the validation loss, the logit sensitivity and active embedding norms against naive replay and AdamAR. While UWSR is able to maintain a low logit sensitivity and improvements in validation loss beyond one epoch, it does not place any direct restrains on the embeddings, allowing free grownth of the embedding norms as shown.



(a) Avazu **(b) Amazon Electronics** **(c) MovieLens-20M**

Figure 3: Validation loss and uncertainty-weighted logit sensitivity ($s_{F,i}(x)^\top \widehat{\Sigma}_{F,i} s_{F,i}(x)$) measured via empirical fisher approximation (Appendix B.3) and norm of active embeddings each training step on Avazu, Amazon Electronics, and MovieLens-20M for a single seed. When comapred to baseline methods, UWSR is able to control the logit sensitivity, while allowing free growth of embedding norms and show improvements beyond one epoch.

6 Conclusion

Multi-epoch training in sparse-embedding recommendation models suffers from self-influence asymmetry: later passes score examples through rows containing target leakage written by their own earlier updates. Our scalar model, local row analysis, embedding-consumer update order intervention show that this prequential breakdown creates an incentive for the shared consumer to rely on rarely occurring rows. To resolve this without direct interventions on embeddings, we propose uncertainty-weighted sensitivity regularization (UWSR), which uses double backpropagation weighted by frozen inverse row information to constrain target sensitivity on poorly determined rows.

Our analysis is rowwise and assumes independent embedding updates, meaning it does not model cross-row interactions. Additionally, the formulation considers fixed-window replay, so adapting the method to continual learning would require dynamically updating the row information matrices as new independent examples arrive. Finally, while our embedding-consumer update order intervention demonstrates that within a minibatch, self-inclusion is sufficient to trigger target sensitivity escalation, further work is needed to measure its exact contribution relative to other factors driving the overall multi-epoch degradation.

AI Use Disclosure

In this work, we used generative AI tools to assist with editing and refining the academic writing, as well as to help draft and structure the source code for the algorithms. We did not use generative AI tools to formulate the core scientific concepts, mathematical claims, or experimental designs. The authors have carefully reviewed all AI-assisted content and take full responsibility for it.

Reproducibility Statement

To ensure the complete reproducibility of our experiments, all datasets and model architectures utilized in this work are entirely publicly available. Detailed descriptions of the experimental setup, data pre-processing steps, and hyperparameter configurations have been included. The source code implementing our algorithms and training pipelines will be made publicly available upon publication.

Ethics Statement

The research presented in this paper focuses on foundational machine learning methodologies. Our experiments rely exclusively on standard, publicly available datasets, and the work does not involve human subjects, personally identifiable information, or sensitive topics. To the best of our knowledge, there are no immediate ethical concerns or direct risks of negative societal impact arising from the specific algorithms and results discussed in this work.

References

Chris M Bishop. Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7 (1):108–116, 1995.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pp. 1613–1622. PMLR, 2015.

Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James Robins. Double/debiased machine learning for treatment and structural parameters, 2018.

A Philip Dawid. Present position and potential developments: Some personal views statistical theory the prequential approach. *Journal of the Royal Statistical Society: Series A (General)*, 147(2): 278–290, 1984.

Harris Drucker and Yann Le Cun. Improving generalization performance using double backpropagation. *IEEE transactions on neural networks*, 3(6):991–997, 1992.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Bradley Efron. The estimation of prediction error: covariance penalties and cross-validation. *Journal of the American Statistical Association*, 99(467):619–632, 2004.

Zhongxiang Fan, Zhaocheng Liu, Jian Liang, Dongying Kong, Han Li, Peng Jiang, Shuang Li, and Kun Gai. Multi-epoch learning with data augmentation for deep click-through rate prediction. *arXiv preprint arXiv:2407.01607*, 2024.

F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114 (13):3521–3526, 2017.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pp. 1885–1894. Pmlr, 2017.

Mang Li and Wei Lyu. Adaptive regularization for large-scale sparse feature embedding models. In *International Conference on Learning Representations*, volume 2026, pp. 44098–44118, 2026.

Yu Xin Li, Felix Dangel, Derek Tam, and Colin Raffel. Fishers for free? Approximating the Fisher information matrix by recycling the squared gradient accumulator. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 34252–34270. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/li25j.html.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pp. 2408–2417. PMLR, 2015.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pp. 43–52, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450336215. doi: 10.1145/2766462.2767755. URL https://doi.org/10.1145/2766462.2767755.

Daniele Micci-Barreca. A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. *ACM SIGKDD explorations newsletter*, 3(1):27–32, 2001.

Adrian Pagan. Econometric issues in the analysis of regressions with generated regressors. *International economic review*, 25(1):221–247, 1984.

Daryl Pregibon. Logistic regression diagnostics. *The annals of statistics*, 9(4):705–724, 1981.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pp. 1161–1170, 2019.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html.

Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. Dcn v2: Improved deep & cross network and practical lessons for web-scale learning to rank systems. In *Proceedings of the web conference 2021*, pp. 1785–1797, 2021.

Steve Wang and Will Cukierski. Click-through rate prediction. https://www.kaggle.com/competitions/avazu-ctr-prediction, 2014. Kaggle.

Buyun Zhang, Liang Luo, Yuxin Chen, Jade Nie, Xi Liu, Shen Li, Yanli Zhao, Yuchen Hao, Yantao Yao, Ellie Dingqiao Wen, Jongsoo Park, Maxim Naumov, and Wenlin Chen. Wukong: Towards a scaling law for large-scale recommendation. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 59421–59434. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/zhang24ao.html.

Zhao-Yu Zhang, Xiang-Rong Sheng, Yujing Zhang, Biye Jiang, Shuguang Han, Hongbo Deng, and Bo Zheng. Towards understanding the overfitting phenomenon of deep click-through rate models. In *Proceedings of the 31st ACM international conference on information & knowledge management*, pp. 2671–2680, 2022.

Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pp. 1059–1068, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450355520. doi: 10.1145/3219819.3219823. URL https://doi.org/10.1145/3219819.3219823.

A Proof of Proposition 1

Proof of Proposition 1. Condition on the pre-update reference state $\mathcal{G}_{t^-}$, $\mathcal{I}^-_{F,i}$, $s_{F,i,t}$, and $\hat{y}_t$ are deterministic. For a displacement δ to row $e_{F,i}$, let $\widetilde{Q}(\delta)$ be a local quadratic approximation for the change in the loss function. Thus,

$$\widetilde{Q}(\delta) = \tfrac{1}{2}\delta^{\mathsf{T}}\mathcal{I}^-_{F,i}\delta - r_t s^{\mathsf{T}}_{F,i,t}\delta + \tfrac{1}{2}h_t(s^{\mathsf{T}}_{F,i,t}\delta)^2.$$

From first-order optimality condition on $\widetilde{Q}(\delta)$, $\mathcal{I}_{F,i}\delta = r_t s_{F,i,t}$, gives $\delta e^{\text{self}}_{F,i,t} = r_t\mathcal{I}^{-1}_{F,i}s_{F,i,t}$ and $u_t = r_t\|s_{F,i,t}\|^2_{\mathcal{I}^{-1}_{F,i}}$, completing proof for part one.

Defining $a_t = \|s_{F,i,t}\|^2_{(\mathcal{I}^-_{F,i})^{-1}}$, the Sherman-Morrison identity gives

$$\|s_{F,i,t}\|^2_{\mathcal{I}^{-1}_{F,i}} = \frac{a_t}{1+h_t a_t} \qquad \text{and} \qquad h_t\|s_{F,i,t}\|^2_{\mathcal{I}^{-1}_{F,i}} = \frac{h_t a_t}{1+h_t a_t} \in [0,1).$$

As BCE loss is affine in its label, $\ell(v,y) = \log(1+e^v) - yv$. Consequently, for any u,

$$\mathbb{E}\left[\ell(z_t+u,y_t) - \ell(z_t+u,y'_t) \mid \mathcal{G}_{t^-}, x_t, y_t\right] - \mathbb{E}\left[\ell(z_t,y_t) - \ell(z_t,y'_t) \mid \mathcal{G}_{t^-}, x_t, y_t\right] = -(y_t-\hat{y}_t)u.$$

Substituting $u = u_t$ proves $D_{F,i,t} = -r_t^2\|s_{F,i,t}\|^2_{\mathcal{I}^{-1}_{F,i}}$, and the pointwise bound follows from $|r_t| \le 1$. Under local correct specification $\hat{y}_t = \hat{y}^\star_t$, $\mathbb{E}[r_t^2 \mid \mathcal{G}_{t^-}] = \hat{y}^\star_t(1-\hat{y}^\star_t) = h_t$, which yields the expected identity in part 2. □

B UWSR Construction

B.1 Method details

For sigmoid BCE, the rowwise expected Fisher and generalized Gauss–Newton information coincide. We instantiate their rowwise scale with an AdaGrad-style cumulative gradient-energy statistic (Duchi et al., 2011). Let $g^B_{F,i}$ denote the row gradient of the minibatch loss for row (F,i) of table F, whose embeddings have dimension d_F:

$$\widehat{\mathcal{I}}_{F,i} := \sum_{B:\, i\in\mathcal{A}(B)} \frac{\|g^B_{F,i}\|_2^2}{d_F}, \qquad \widetilde{\mathcal{I}}_{F,i} := \begin{cases} (1-\gamma_{\text{shr}})\widehat{\mathcal{I}}_{F,i} + \gamma_{\text{shr}}\,\overline{\mathcal{I}}_F, & i \in \mathcal{O}, \\ \overline{\mathcal{I}}_F, & i \notin \mathcal{O}, \end{cases} \tag{8}$$

$$\epsilon_F := 10^{-6}\overline{\mathcal{I}}_F, \qquad \rho_{F,i} := \frac{1}{\max\{\widetilde{\mathcal{I}}_{F,i}, \epsilon_F\}},$$

$$c := \frac{1}{|\mathcal{O}|}\sum_{(F,i)\in\mathcal{O}} \rho_{F,i}, \qquad \widehat{\Sigma}_{F,i} := \frac{\rho_{F,i}}{c} I, \tag{9}$$

where $\mathcal{O}$ is the set of rows observed during the first pass, $\overline{\mathcal{I}}_F$ is the mean accumulated information among observed rows of table F, and γ_{shr} is a one-time table-level shrinkage coefficient. The accumulator has no exponential decay or bias correction: inactive rows simply receive zero additional information. The resulting weights are frozen after the first pass.

The minibatch construction preserves the Fisher/GGN scale needed here. Let $g_{F,i,t}$ be occurrence t's row gradient. Writing $g^B_{F,i} = |B|^{-1}\sum_{t\in B_i} g_{F,i,t}$, the usual Fisher score identity $\mathbb{E}[g_{F,i,t} \mid x_t] = 0$ and conditional independence make cross-occurrence terms vanish, so

$$\mathbb{E}\big[\|g^B_{F,i}\|_2^2 \mid B_i\big] = |B|^{-2}\sum_{t\in B_i}\mathbb{E}\big[\|g_{F,i,t}\|_2^2 \mid x_t\big]. \tag{10}$$

The fixed batch-reduction factor is removed by the normalization in Equation (9); accumulation across batches measures information along the same first-pass trajectory that fits the rows.

The scalarization has a direct optimizer analogue. UWSR uses cumulative row-wise AdaGrad accumulation in time; the same one-scalar-per-row reduction is also common in row-wise Adam

implementations. UWSR reuses the AdaGrad statistic in a dual role: the optimizer scales a row update by its inverse square root, whereas UWSR weights downstream sensitivity by its inverse because it is propagating row uncertainty:

$$\Delta e_{F,i}^{\mathrm{AdaGrad}} \propto -\widehat{\mathcal{I}}_{F,i}^{-1/2} g_{F,i}, \qquad R_{F,i} \propto \left\|\widehat{\mathcal{I}}_{F,i}^{-1/2} s_{F,i}\right\|_2^2.$$

A per-coordinate UWSR construction gave essentially identical performance; Appendix F reports the comparison.

We also evaluate UWSR Expected Fisher, which replaces the realized gradient second moment with the occurrence-level conditional moment $\hat{y}_t(1-\hat{y}_t)\|s_{F,i}(x_t)\|_2^2/d_F$. A uniform ablation sets $\widehat{\Sigma}_{F,i} = I$.

The penalty requires second-order autodifferentiation through $s_{F,i}(x_t)$ (for example, `create_graph=True`). The variants differ only in how they estimate row uncertainty and have nearly identical computational cost. Their close agreement in Table 2 shows that the result is insensitive to choosing the realized or conditional score moment. Table **??** reports the latter's table-level information and normalized inverse weights;

B.2 UWSR Training Procedure

Algorithm 1 UWSR with a row-wise empirical-Fisher accumulator

Require: Ordered minibatches $\mathcal{D}$; epochs K; consumer parameters θ; embedding tables $\{E_F\}$; penalty strength λ; shrinkage γ_{shr}

Initialize $S_{F,i} \leftarrow 0$ for every embedding row
for $q = 1, \ldots, K$ **do**
  for minibatch $B \in \mathcal{D}$ **do**
    Retrieve occurrence-level lookup outputs $a_{b,F,i} = E_F[i]$ for $(F,i) \in \mathcal{A}(x_b)$
    Compute logits $z_b = z_\theta(x_b)$ and $L_{\mathrm{BCE}} = |B|^{-1} \sum_{b \in B} \ell(z_b, y_b)$
    if $q = 1$ **then**
      for all rows (F,i) activated by B **do**
        $g_{B,F,i} \leftarrow \nabla_{E_F[i]} L_{\mathrm{BCE}}$
        $S_{F,i} \leftarrow S_{F,i} + \mathrm{stopgrad}\big(\|g_{B,F,i}\|_2^2/d_F\big)$
      end for
      $L \leftarrow L_{\mathrm{BCE}}$
    else
      for all $b \in B$ and $(F,i) \in \mathcal{A}(x_b)$ **do**
        $s_{b,F,i} \leftarrow \nabla_{a_{b,F,i}} z_b$ ▷ retain the derivative graph
      end for
      $R_B \leftarrow |B|^{-1} \sum_{b \in B} \sum_{(F,i) \in \mathcal{A}(x_b)} u_{F,i} \|s_{b,F,i}\|_2^2$
      $L \leftarrow L_{\mathrm{BCE}} + \lambda R_B$
    end if
    Update θ and $\{E_F\}$ using ∇L
  end for
  if $q = 1$ **then**
    for all embedding tables F **do**
      $\overline{\mathcal{I}}_F \leftarrow |\mathcal{O}_F|^{-1} \sum_{i \in \mathcal{O}_F} S_{F,i}$
      for all rows i in table F **do**
        $\widetilde{\mathcal{I}}_{F,i} \leftarrow (1-\gamma_{\mathrm{shr}}) S_{F,i} + \gamma_{\mathrm{shr}} \overline{\mathcal{I}}_F$ if $i \in \mathcal{O}_F$
        $\widetilde{\mathcal{I}}_{F,i} \leftarrow \overline{\mathcal{I}}_F$ otherwise
        $\rho_{F,i} \leftarrow \max\{\widetilde{\mathcal{I}}_{F,i}, 10^{-6} \overline{\mathcal{I}}_F\}^{-1}$
      end for
    end for
    $c \leftarrow |\mathcal{O}|^{-1} \sum_{(F,i) \in \mathcal{O}} \rho_{F,i}$
    $u_{F,i} \leftarrow \rho_{F,i}/c$ for every row; freeze all $u_{F,i}$
  end if
end for

B.3 ALTERNATIVE Σ CONSTRUCTIONS

Table-Level Shrinkage and Weight Normalization: For numerical stability during practical experiments, a shrink was applied. We have not found it sensitive in informal ablations. Differing non-semantically, the scale of all weight coefficients is normalized such that tuned λ will be closer to 1.

Expected-Fisher variant. The expected-Fisher variant accumulates the occurrence-level quantity $p_t(1-p_t)\|s_{F,i}(x_t)\|_2^2/d_F$ before row aggregation, where p_t is the model's prediction for sample t. It uses the same shrinkage, normalization, and later-pass objective as standard UWSR.

Uniform sensitivity (unweighted ablation). $\widehat{\Sigma}_{F,i} = I$, with any shared scalar absorbed into λ. This is the unweighted embedding-row score-sensitivity penalty and carries no row- or table-specific information. Its squared-derivative form is related to double backpropagation and input-Jacobian regularization Drucker & Le Cun (1992).

C DATASET DETAILS

MovieLens-20M. All 20,000,263 ratings, predicting whether a rating is at least four stars. The examples are sorted by timestamp and split into contiguous 80% training and 5% validation, calibration, monitoring, and held-out windows, over 138,493 users and 26,744 items. Inputs are user and item identifiers, 19 genres, release year, training-only support, availability indicators, and 1,128 genome-tag relevance features. In the monitoring window approximately 85% of examples have a user unseen in training and 17% an unseen item.

AmazonElectronics. The official training file provides 2,608,764 binary examples; the official test file is divided into validation, calibration, monitoring, and held-out partitions of approximately 96,200 each. Inputs are user, target item, target category, and mean-pooled item and category histories truncated to length 50, over 192,403 users, 63,001 items and 801 categories. Monitoring users and items all occur in training.

Avazu. All 40,428,967 binary click examples in the RecZoo preprocessing. The official training file provides 28,300,276 examples; the labeled evaluation files are concatenated in original order and divided into validation, calibration, monitoring, and held-out partitions of approximately 3.03 million each. Each of the 22 categorical fields has a separate embedding table, with cardinalities from 5 to 1,048,284, and evaluation-time unseen rates vary by field and reach 34.6%.

D MODEL, TRAINING AND HYPERPARAMETER DETAILS

Model Architecture. Sparse tables use 32-dimensional embeddings. The shared consumer is a WuKong-inspired interaction model Zhang et al. (2024) containing three residual interaction blocks, four rank-16 factorization heads in each block, followed by a 128-unit SiLU output network.

Model Initialization. Sparse tables are zero-initialized, with symmetry broken by the linear projection path of the interaction block; the factorization heads contribute no embedding gradient at exact zero.

Hyperparameter search. The learning rate is 2×10^{-3} and the batch size is 4,096. We select penalty coefficients using the validation partition at $\sqrt{(}10)$ interval spacing.

Table 4: Paired test-set changes at epoch four, averaged over five seeds. Brackets give two-sided 95% t-intervals across paired seed-level differences from the corresponding one-pass Adam checkpoint; positive values are improvements.

Method	MovieLens	Amazon	Avazu
(a) Relative ΔBCE (%) with 95% CI			
Naive replay	$-33.66\ [-45.12, -22.20]$	$-95.43\ [-102.76, -88.10]$	$-13.85\ [-15.57, -12.14]$
MEDA	$+0.21\ [-2.37, +2.79]$	$+3.92\ [+3.23, +4.62]$	$-0.11\ [-0.58, +0.37]$
Freeze until final pass	$-0.90\ [-3.60, +1.81]$	$-1.74\ [-3.07, -0.42]$	$-0.20\ [-0.81, +0.40]$
Active-row AdamW	$+0.10\ [-2.45, +2.65]$	$-23.54\ [-25.58, -21.51]$	$-5.76\ [-7.19, -4.34]$
Embedding dropout	$-2.63\ [-4.36, -0.90]$	$-7.16\ [-8.82, -5.50]$	$-7.77\ [-10.72, -4.82]$
AdamAR	$-0.86\ [-4.81, +3.10]$	$+2.92\ [+2.08, +3.75]$	$+1.46\ [+0.86, +2.05]$
UWSR Uniform	$+1.19\ [-1.39, +3.78]$	$-1.21\ [-1.84, -0.58]$	$+0.48\ [-0.19, +1.15]$
UWSR	$+1.38\ [-0.89, +3.64]$	$+6.78\ [+6.17, +7.40]$	$+2.29\ [+1.66, +2.92]$
UWSR Expected Fisher	$+1.18\ [-1.36, +3.73]$	$+6.93\ [+6.41, +7.46]$	$+2.49\ [+1.90, +3.08]$
(b) ΔAUC with 95% CI			
Naive replay	$-0.1669\ [-0.1879, -0.1458]$	$-0.0290\ [-0.0323, -0.0258]$	$-0.0568\ [-0.0635, -0.0501]$
MEDA	$-0.0060\ [-0.0265, +0.0146]$	$+0.0126\ [+0.0084, +0.0168]$	$+0.0008\ [-0.0032, +0.0048]$
Freeze until final pass	$-0.0207\ [-0.0450, +0.0037]$	$-0.0063\ [-0.0112, -0.0015]$	$-0.0004\ [-0.0031, +0.0023]$
Active-row AdamW	$+0.0010\ [-0.0201, +0.0222]$	$-0.0712\ [-0.0742, -0.0682]$	$-0.0214\ [-0.0245, -0.0184]$
Embedding dropout	$-0.0407\ [-0.0591, -0.0222]$	$-0.0106\ [-0.0124, -0.0089]$	$-0.0458\ [-0.0577, -0.0339]$
AdamAR	$-0.0014\ [-0.0193, +0.0165]$	$+0.0133\ [+0.0093, +0.0174]$	$+0.0096\ [+0.0064, +0.0128]$
UWSR Uniform	$+0.0054\ [-0.0209, +0.0317]$	$-0.0007\ [-0.0035, +0.0021]$	$+0.0002\ [-0.0044, +0.0047]$
UWSR	$+0.0058\ [-0.0165, +0.0281]$	$+0.0231\ [+0.0195, +0.0267]$	$+0.0141\ [+0.0121, +0.0162]$
UWSR Expected Fisher	$+0.0030\ [-0.0276, +0.0335]$	$+0.0232\ [+0.0197, +0.0268]$	$+0.0154\ [+0.0126, +0.0182]$

E Validation Results and Additional Comparisons

Table 3: Validation ΔBCE (%) at epoch four for algorithms compared in Table 2. Positive values denote improvement over the corresponding one-pass model.

Method	MovieLens	Amazon	Avazu
Naive replay	-24.66	-101.03	-15.71
MEDA	-0.93	$+3.37$	-0.06
Freeze embeddings	-1.89	-1.45	$+0.06$
AdamAR	-0.65	$+2.19$	$+1.96$
UWSR Uniform sensitivity (unweighted ablation)	$+0.40$	-1.05	$+2.50$
UWSR, rowwise	$+1.47$	$+6.77$	$+3.97$
UWSR Expected Fisher, rowwise	$+1.43$	$+6.95$	$+4.11$

F Accumulator granularity ablation

Table 5: Accumulator-granularity ablation on Amazon Electronics after four epochs (seed 7). Both variants selected $\lambda = 0.03$ on validation. Auxiliary state excludes model parameters and optimizer state.

Accumulator	Scalars per row	λ	Validation BCE ↓	AUC ↑
Row-wise scalar	1	0.03	**0.442510**	**0.876256**
Per-coordinate diagonal	32	0.03	0.442851	0.875963

G ARCHITECTURE RESULTS

Table 6: Amazon validation results across WuKong, DCNv2 (Wang et al., 2021), and AutoInt (Song et al., 2019). Entries are ΔBCE relative to each architecture's one-pass baseline.

	One pass	ΔBCE						Fisher
Architecture	BCE	Naive	MEDA	Freeze	AdamAR	Isotropic	UWSR	ΔAUC
WuKong	0.474634	−101.03	+3.37	−1.45	+2.19	−1.05	**+6.95**	+0.0222
DCNv2	0.507843	−45.23	+6.71	+3.08	+5.90	+2.26	**+9.02**	+0.0362
AutoInt	0.491371	−57.86	+5.29	−3.35	+5.56	−0.97	**+8.58**	+0.0303

H COMPUTATIONAL COST

Table 7: Amazon computational overhead relative to naive replay four-pass training, mean 2 seeds. Later-epoch step time is relative to one naive replay step; end-to-end overhead includes all epochs and method-specific setup.

Method	Epochs	Later-epoch step time	End-to-end overhead
Naive replay / MEDA / AdamAR	4	1.00×	1.00×
UWSR	4	1.18×	1.13×

The paired arms ran sequentially on otherwise idle GPUs with diagnostics and candidate caching disabled. Per-seed later-pass ratios are 1.16 and 1.20; end-to-end ratios are 1.12 and 1.14.