FUMO: FREQUENCY UNIFIED EMBEDDING OPTIMIZATION FOR SEQUENTIAL RECOMMENDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Sequential recommendation systems suffer from highly imbalanced item frequency distribution in real-world datasets. This imbalance can contribute to popularity bias and significantly influences recommendation accuracy. Training trajectories show that frequent item accuracy plateaus early while rare items require continued training for accuracy improvement, revealing unequal learning progress under the shared training schedule. To solve this problem, we propose Frequency Unified eMbedding Optimization (FUMO), an optimizer that uses item frequency to adjust recommendation updates for item embeddings. FUMO applies larger updates for recommendation loss on rare items and combines frequency-dependent weight decay with adaptive shrinkage to control the overfitting, without modifying the model architecture or introducing additional loss functions. Across six sequential recommendation models and four datasets, FUMO shows compelling performance compared with other baseline optimizers and improves HR@20 up to 18.5% and NDCG@20 up to 25.0% over Adam. Training trajectories additionally show earlier improvements in rare item validation. These results support item frequency as a useful signal for adjusting embedding optimization across diverse sequential recommendation systems, and show that FUMO is a competitive optimizer for sequential recommendation.

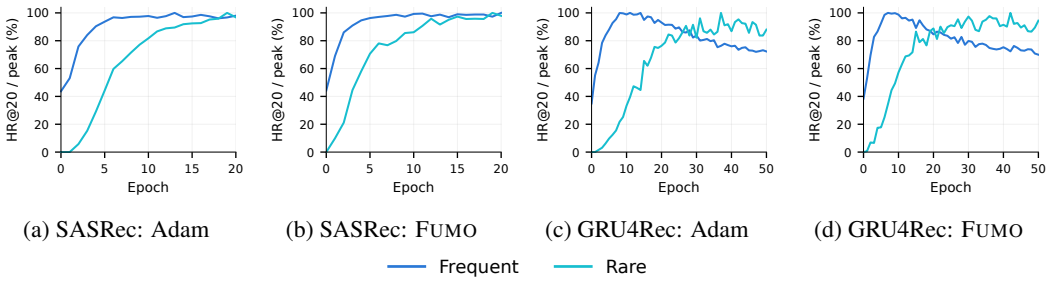
1 INTRODUCTION

Sequential recommendation predicts a user's next interaction from their interaction history (Hidasi et al., 2016; Kang & McAuley, 2018; Sun et al., 2019). These interaction frequencies are typically highly imbalanced: a small fraction of items accounts for many interactions, while most items appear rarely. This imbalance is also known as popularity bias and makes learning reliable representations for rare items difficult. Models trained on imbalanced items favor frequent items and under-recommends rare ones (Steck, 2011; Klimashevskaia et al., 2024), which limits recommendation accuracy. At the same time, popularity can reflect genuine user preferences as well as unequal exposure, so suppressing popular items may discard useful recommendation signals (Zhao et al., 2023). Improving rare-item recommendations therefore requires understanding how models learn from uneven observations.

We examine this problem through the training process as frequent and rare items may require different amounts of training iterations to develop useful representations. The training analyses reveal a gap in learning progress: as illustrated in Figure 1, frequent item accuracy rises rapidly and often plateaus early, while rare item accuracy improves more slowly and continues increasing later in training. Extending training can provide additional gains for rare items, but frequent item accuracy might decline while rare item accuracy is still improving. Selecting a single training stop time that accommodates groups whose useful learning occurs at different stages is difficult. These observations motivate adjusting how embeddings are updated so that rare items can make useful progress earlier within the shared training schedule. Prior work has connected imbalanced interactions to unequal gradient contributions, and adaptive optimizers such as Adam (Kingma & Ba, 2015) could alleviate this issue (Ren et al., 2022), but the observed trajectories show that this adaptation does not fully eliminate the gap in learning progress.

Existing studies on recommendation systems mainly target the popularity bias directly through exposure correction, representation learning, auxiliary objectives, and re-ranking (Schnabel et al., 2016; Gupta et al., 2019; Abdollahpouri et al., 2019; Qiu et al., 2022), which does not fully solve

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.



(a) SASRec: Adam (b) SASRec: FUMO (c) GRU4Rec: Adam (d) GRU4Rec: FUMO

Figure 1: Learning progress of frequent (top 20% items) and rare items (bottom 80% items) under optimizer Adam (a)(c) and optimizer FUMO (b)(d), for SASRec and GRU4Rec on Sports dataset. Each curve shows validation HR@20 normalized by peak accuracy. FUMO narrows the progress gap by bringing rare items closer to their peak accuracy earlier in training.

this issue. On the other hand, frequency-aware optimization provides another perspective. FA-SGD (Li et al., 2022) assigns larger learning rates to infrequent embeddings and establishes convergence benefits in a sparse embedding setting, while FAL (Qiu et al., 2025) instead reduces learning rates for infrequent embeddings to mitigate multi-epoch overfitting in advertising models. These approaches highlight the difficulty for item embedding learning: rare items make slow progress during training yet their limited observations also make overfitting a concern. The delayed improvements of rare items motivate us to use larger relative recommendation updates with explicit regularization and overfitting control. Moreover, some models supplement the recommendation loss with auxiliary objectives (Qiu et al., 2022; Yang et al., 2026). We aim to solve this problem that distinguishes recommendation gradients without introducing additional loss.

We propose **Frequency Unified eMbedding Optimization** (FUMO), a novel frequency-aware optimizer for item embeddings. FUMO assigns each item a recommendation-update multiplier that decreases with its frequency, giving rarer items larger relative step sizes. The multiplier is applied to the recommendation gradient in the first moment only, while retaining auxiliary gradients and second moment at their base coefficient. This optimizer design preserves the intended frequency scaling through Adam's normalization and avoids directly multiplying auxiliary contributions by the same frequency factor. To prevent rare item from overfitting, FUMO applies frequency-dependent weight decay coefficients and adaptive shrinkage base on the second moment (Ding et al., 2025) . Recommendation scaling and shrinkage thus provide separate controls over learning progress and embedding growth and overfitting. FUMO is directly applicable to various model architectures and does not introduce new loss functions.

Our contributions are as follows: (1) **Characterizing unequal learning progress.** We examine training trajectories by item frequency and identify a recurring pattern in which frequent-item accuracy plateaus while rare-item accuracy continues improving. This observation motivates improving rare-item progress within a shared training schedule. (2) **Frequency unified optimization.** We develop FUMO, a novel optimizer for sequential recommendation that combines frequency-aware recommendation updates and adaptive weight decay that not only boosts rare items learning progress but also controls the rare item overfitting issue. FUMO requires no architectural changes or additional losses. (3) **Empirical experiments.** We conduct experiments across six sequential recommendation models, including models with contrastive and regularization objectives, on four real-world datasets. FUMO demonstrates consistent accuracy improvement in all model–dataset combinations over Adam, up to 18.5% in HR@20 and 25.0% in NDCG@20. FUMO also achieves the highest or second-highest recommendation accuracy among other baseline optimizers. Training trajectories also show FUMO's effectiveness in earlier rare item accuracy gains. These results support item frequency as a useful signal for adjusting embedding optimization across diverse sequential recommendation systems, and show that FUMO is a competitive optimizer for sequential recommendation.

2 Related Work

Sequential Recommendation System: Sequential recommendation has been proposed to predict future interactions based on a user's interaction history. Early models include Markov chains (Rendle et al., 2010), convolutional neural networks (Tang & Wang, 2018), and recurrent neural networks (Hidasi et al., 2016). Transformers and attention-based models (Kang & McAuley, 2018; Sun et al., 2019) use self-attention to capture dependencies across sequence positions and achieve superior performance. Contrastive learning models assist item representation learning through augmented views (Zhou et al., 2020; Xie et al., 2022; Qiu et al., 2022). More recently, frequency-domain models have been proposed to suppress the high-frequency noises in the sequences via learnable filters (Zhou et al., 2022; Shin et al., 2024). Here, frequency refers to spectral components of a sequence, rather than item occurrence counts. Despite these advances in sequence modeling, these sequential recommendation systems remain vulnerable to popularity bias, and learning from imbalanced interaction histories can favor frequent items and leave rare items under-recommended regardless of model architecture (Klimashevskaia et al., 2024), while improving the sequence encoders does not explicitly correct this bias. Hence, we need to take extra effort to target the popularity bias.

Mitigating Popularity Bias in Recommendation: Existing methods on popularity bias primarily modify the model, training objective, or recommendation output. Methods include normalization (Gupta et al., 2019) and counter-factual modeling (Wei et al., 2021), while representation disentanglement (Zheng et al., 2021; Liu & Zheng, 2020; Zhang et al., 2023) allows rare item learning to be separated from frequent items. These approaches modify the model or its representation structure, but do not directly address how popularity bias emerges during training. FUMO targets this process by modifying embedding updates within the existing architecture. Re-ranking (Abdollahpouri et al., 2019), calibration (Steck, 2018), and test-time embedding adjustment (Ren et al., 2022) modify recommendation results after training. These methods are applied only at evaluation or test time when parameters are finalized and cannot assist the training phase. FUMO directly adjusts the embedding updates during training and allows rare items embeddings to learn better features before the training terminates.

Other methods mitigate popularity bias by modifying the training objective. Long-tail classification method adjusts the contribution of observed interactions (Saito et al., 2020; Cao et al., 2019; Menon et al., 2021; Alshammari et al., 2022), while spectral regularization constrains transformations that amplify popularity differences (Lin et al., 2025; Yang et al., 2026). Contrastive learning introduces self-supervised signals to improve representations (Yu et al., 2022; Cai et al., 2024) and prevents representation degeneration associated with popularity bias (Qiu et al., 2022). However, these additional signals can introduce their own difficulties: false negatives may push semantically similar examples apart (Chuang et al., 2020). Combining multiple objectives may also create conflicting gradients, as improving one objective can hinder another and degrade joint optimization (Yu et al., 2020). FUMO modifies the optimization process without modifying the objective, allowing it to complement these methods.

Frequency-Aware Optimization: AdaGrad (Duchi et al., 2011) and Adam (Kingma & Ba, 2015) adapt coordinate-wise steps using gradient moments, and rarely updated coordinates may receive larger updates. Though Adam has been widely used as the default optimizer in many machine learning domains, the moments are still implicit reflections of the popularity without explicitly considering the item frequency. Kunstner et al. (2024) demonstrates that Adam could alleviate the problem of rare tokens making little progress in SGD, but the gap of progress still exists. We will further demonstrate this gap in Section 3.2.

Frequency-aware optimizers directly manipulate the item learning rate based on item frequency. FA-SGD (Li et al., 2022) assigns larger steps to infrequent tokens and establishes convergence benefits under sparse embedding updates. Its reported results match or improve on adaptive optimizers in sparse embedding tasks, with substantial memory savings, but do not establish a consistent accuracy advantage over AdaGrad and Adam (Li et al., 2022). FAL (Qiu et al., 2025) takes the opposite approach of reducing learning rates for infrequent embeddings to mitigate multi-epoch over-fitting problem, which does not target the popularity bias. FUMO directly addresses the slow progress of learning the embedding of rare items through larger steps sizes. Furthermore, FUMO applies frequency-dependent weight decay to handle the same over-fitting issue. Helen (Zhu et al., 2024) was develops for click-through rate prediction and uses sharpness-aware minimization (SAM) (Foret et al.,

2021) with embedding perturbations scaled by normalized feature frequency. Stronger perturbations for frequent features target their larger Hessian eigenvalues. Its update evaluates gradients before and after perturbation, whereas FUMO uses one gradient evaluation per step. We will compare the performance and recommendation accuracy of different optimizers in Section 5.2.

3 PRELIMINARIES

3.1 SEQUENTIAL RECOMMENDATION WITH A TIED ITEM TABLE

Sequential Recommendation Problem and Dataset. Let $\mathcal{U}$ be the set of users and $\mathcal{I} = \{1, \dots, M\}$ be the item set with M items. Each user u has an item interaction history denoted by a sequence $< v_{u,1}, v_{u,2}, \dots, v_{u,L_u} >$ where $L_u \geq 2$ is the number of interactions by user u and $v_{u,s} \in \mathcal{I}$ for $s \in [1, L_u]$. For each item $i \in \mathcal{I}$, let $n_i = \sum_{u \in \mathcal{U}} \sum_{s=1}^{L_u} \mathbf{1}_{v_{u,s}=i}$ be the item frequency, which is the total number of occurrences of item i in users' interaction histories. Sequential recommendation uses the last item $y_u = v_{u,L_u}$ as the prediction target and previous interaction sequence $\boldsymbol{x}_u =< v_{u,1}, v_{u,2}, \dots, v_{u,L_u-1} >$ as model input.

Model. Recommendation system includes an item embedding layer and a sequential encoder. The item embedding layer $E \in \mathbb{R}^{M \times d}$ gives a d-dimensional vector representation $\boldsymbol{e}_i$ for each item i. Consider user u. The sequential encoder f_θ with parameters θ, taking the previous user interaction $\boldsymbol{x}_u$ and the item embedding layer E as input, computes the hidden representation $\boldsymbol{h}_u \in \mathbb{R}^d$. That is, $\boldsymbol{h}_u = f_\theta(\boldsymbol{x}_u; E)$. The predicted probability distribution for target item i is the softmax over the full item catalog,

$$p_{u,i} = p(i \mid \boldsymbol{x}_u) = \frac{\exp\left(\boldsymbol{e}_i^\top \boldsymbol{h}_u\right)}{\sum_{j=1}^{M} \exp\left(\boldsymbol{e}_j^\top \boldsymbol{h}_u\right)} \tag{1}$$

Objective. The training objective consists of two losses: recommendation loss $\mathcal{L}_{\text{rec}}$ and auxiliary loss $\mathcal{L}_{\text{aux}}$.

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{aux}}, \qquad \mathcal{L}_{\text{rec}}(\boldsymbol{x}_u, y_u) = -\log p(y_u \mid \boldsymbol{x}_u). \tag{2}$$

$\mathcal{L}_{\text{rec}}$ is usually defined as cross-entropy loss and measures how well the model gives predictions. $\mathcal{L}_{\text{aux}}$ provides additional regularization or constraints. We regard it as the sum of all other losses that is non-cross-entropy recommendation loss as a *single* loss. This term $\mathcal{L}_{\text{aux}}$ is optional and depends on model architecture and other recommendation techniques such as self-supervised contrastive learning (Zhou et al., 2020; Xie et al., 2022; Qiu et al., 2022). Traditional optimizers directly operate on the summed gradient $\nabla_{\boldsymbol{e}_i} \mathcal{L} = \nabla_{\boldsymbol{e}_i} \mathcal{L}_{\text{rec}} + \nabla_{\boldsymbol{e}_i} \mathcal{L}_{\text{aux}}$. The most common choice is L_2 regularization loss, where $\mathcal{L}_{\text{aux}} = \frac{\lambda_{L_2}}{2} \|E\|_F^2$,$\lambda_{L_2} \geq 0$ is the L_2 regularization coefficient, and $\| \cdot \|_F$ is the Frobenius norm (Golub & Van Loan, 2013).

3.2 UNEQUAL LEARNING PROGRESS ACROSS ITEMS

We could compute the gradient of $\mathcal{L}_{\text{rec}}$ from input user u with respect to the item embedding of item i by Equation (1),

$$\nabla_{\boldsymbol{e}_i} \mathcal{L}_{\text{rec}}(\boldsymbol{x}_u, y_u) = \left(p_{u,i} - \mathbb{1}_{y_u=i}\right) \boldsymbol{h}_u + \left(\frac{\partial \boldsymbol{h}_u}{\partial \boldsymbol{e}_i}\right)^\top \sum_{j=1}^{M} \left(p_{u,j} - \mathbb{1}_{y_u=j}\right) \boldsymbol{e}_j \tag{3}$$

The gradient pulls $\boldsymbol{e}_i$ toward the sequence representation $\boldsymbol{h}_u$ if i is the target, and, otherwise, pushes $\boldsymbol{e}_i$ away. Only the first case tells the model where item i belongs, and it occurs only on samples that involve target i. The weight $p_{u,i}$ is the predicted probability of i, which stays small for rare items. The sequential information is encoded in the gradient only when the item i occurs in the training sequence, where $\frac{\partial \boldsymbol{h}_u}{\partial \boldsymbol{e}_i} \neq 0$. Therefore, rare items generally receive smaller gradients and less sequential information than frequent items, which causes the embedding of rare items to be learnt slower than frequent items. Ren et al. (2022) report the same imbalance in recommendation. To demonstrate this, we divide items into a frequent group and a rare group, and plot the normalized validation accuracy during training. Figure 1 follows the two groups through training under Adam. The frequent group reaches 90% of peak accuracy and saturates within the first three to five epochs, at which point the rare group has reached only 10–40%. We also observe that frequent-item accuracy

can decline while rare-item accuracy is still improving, which means that extending training to benefit rare items can come at the expense of frequent item accuracy.

The same phenomenon could also be found in language model training, where the token frequencies also follow an imbalanced distribution. Kunstner et al. (2024) shows that gradient descent quickly decreases the training loss of frequent groups while rare groups make little progress. Li et al. (2022) prove that scaling learning rate by $\frac{1}{\sqrt{n_i}}$ improves the convergence rate of gradient descent.

4 Proposed Method FUMO

4.1 Frequency-Aware Learning Rate

The training trajectories in Section 3.2 show that rare items continue improving after frequent items have largely saturated. This motivates allocating larger relative steps to rare rows within the same training window. Let $r > 0$ control the frequency scale, $\gamma \geq 1$ be the step size multiplier, and e denote Euler's number. For an item with frequency $n \geq 0$, define

$$\phi_r(n) = \frac{\mathrm{e}\ln(n/r+\mathrm{e})}{n/r+\mathrm{e}}, \qquad \rho_i = \gamma\phi_r(n_i), \tag{4}$$

where ρ_i is the recommendation step size multiplier for item i. The function ϕ_r is positive and strictly decreasing on $[0, \infty)$, with $\phi_r(n) \to 0$ when $n \to \infty$. Thus, rare items receive larger multipliers than frequent items. The additional factor e normalizes $\phi_r(0) = 1$ for convenience. Following Qiu et al. (2025), we introduce γ to adjust the step size by recommendation loss. Let $n_{\max} = \max_{i\in\mathcal{I}} n_i$ be the largest item frequency. For fixed γ, the parameter r controls the range of multipliers through $\gamma\phi_r(n_{\max}) \leq \rho_i \leq \gamma$. It also permits consistent scaling across dataset sizes: for any real number $a > 0$, $\phi_{ar}(an_i) = \phi_r(n_i)$. Scaling the counts and r by the same factor a therefore preserves every item's multiplier.

4.2 Decoupled Update Scaling

Frequency correction targets the recommendation loss. Auxiliary objectives, such as self-supervised or contrastive losses, provide additional training signals whose appropriate scale need not follow item frequency. Multiplying the entire embedding update by ρ_i would also rescale those signals. We therefore scale only the recommendation contribution to the first moment.

At iteration $t \geq 1$, let $\boldsymbol{g}_{\mathrm{rec},i}^{(t)} = \nabla_{\boldsymbol{e}_i}\mathcal{L}_{\mathrm{rec}}$ and $\boldsymbol{g}_{\mathrm{aux},i}^{(t)} = \nabla_{\boldsymbol{e}_i}\mathcal{L}_{\mathrm{aux}}$ denote the recommendation and auxiliary gradients for row i, evaluated at the parameters before the update. Their sum $\boldsymbol{g}_i^{(t)} = \boldsymbol{g}_{\mathrm{rec},i}^{(t)} + \boldsymbol{g}_{\mathrm{aux},i}^{(t)}$ is the total loss gradient. All square roots and divisions below are coordinate-wise, and $\odot$ denotes coordinate-wise multiplication. For momentum coefficients $\beta_1, \beta_2 \in [0, 1)$, we maintain the corrected first moment $\widehat{\boldsymbol{m}}_i^{(t)} \in \mathbb{R}^d$ and the corrected second moment $\widehat{\boldsymbol{v}}_i^{(t)} \in \mathbb{R}^d$:

$$\boldsymbol{m}_i^{(t)} = \beta_1\boldsymbol{m}_i^{(t-1)} + (1-\beta_1)\big(\rho_i\boldsymbol{g}_{\mathrm{rec},i}^{(t)} + \boldsymbol{g}_{\mathrm{aux},i}^{(t)}\big), \qquad \widehat{\boldsymbol{m}}_i^{(t)} = \tfrac{\boldsymbol{m}_i^{(t)}}{1-\beta_1^t}, \tag{5}$$

$$\boldsymbol{v}_i^{(t)} = \beta_2\boldsymbol{v}_i^{(t-1)} + (1-\beta_2)\big(\boldsymbol{g}_i^{(t)} \odot \boldsymbol{g}_i^{(t)}\big), \qquad \widehat{\boldsymbol{v}}_i^{(t)} = \tfrac{\boldsymbol{v}_i^{(t)}}{1-\beta_2^t}. \tag{6}$$

where the bias corrections follow Adam (Kingma & Ba, 2015). Auxiliary gradients $\boldsymbol{g}_{\mathrm{aux},i}^{(t)}$ retain their base coefficient in the numerator and contribute to the shared second moment. The second moment uses the unscaled total gradient, which keeps the frequency multiplier from being canceled.

4.3 Frequency-Aware Weight Decay

Accelerating rare items learning progress also highlights the demand for avoiding overfitting. Qiu et al. (2025) identifies infrequent embedding rows as a source of multi-epoch overfitting and observes that larger learning rates can exacerbate this problem. This motivates pairing our larger relative recommendation steps with stronger weight decay for rare items and supporting continued rare item learning while limiting the growth of their embeddings. Though rare item gradients mainly push them away from representation $\boldsymbol{h}_u$ and stabilized by larger weight decay, the latter also suppress the update when the item is interacted by user, and it's difficult to balance these cases with a single decay rate.

To address this issue, we divide the decay step by $\sqrt{\widehat{\boldsymbol{v}}_i^{(t)}}$. When rare items receive large gradients as targets or in input sequences, the second moment $\widehat{\boldsymbol{v}}_i^{(t)}$ increases and decreases the shrinkage strength. This method is also applied by AdamD (Ding et al., 2025). Let $\lambda \geq 0$ be the base weight decay strength, $\eta > 0$ be the base learning rate, $\mathbf{1} \in \mathbb{R}^d$ be all-ones vector, and $\epsilon > 0$ for numerical stability. Define the item weight decay λ_i and its coordinate-wise shrinkage factor $\boldsymbol{s}_i^{(t)} \in [0,1]^d$ for item i by

$$\lambda_i = \lambda \phi_r(n_i), \qquad \boldsymbol{s}_i^{(t)} = \min\left\{ \frac{\eta\lambda_i}{\sqrt{\widehat{\boldsymbol{v}}_i^{(t)}}+\epsilon}, \mathbf{1} \right\}, \tag{7}$$

Rare items and items with smaller second moment receive stronger decay, and clipping ensures that decay alone neither reverses a coordinate's sign nor increases its magnitude. Combining Equations (5), (6), and (7) gives the complete embedding update:

$$\boldsymbol{e}_i^{(t)} = \left(\mathbf{1} - \boldsymbol{s}_i^{(t)}\right) \odot \boldsymbol{e}_i^{(t-1)} - \eta \frac{\widehat{\boldsymbol{m}}_i^{(t)}}{\sqrt{\widehat{\boldsymbol{v}}_i^{(t)}}+\epsilon}. \tag{8}$$

When $\eta\lambda_i \leq \sqrt{\widehat{\boldsymbol{v}}_i^{(t)}} + \epsilon$ in every coordinate, clipping is inactive and Equation (8) can be written as

$$\boldsymbol{e}_i^{(t)} = \boldsymbol{e}_i^{(t-1)} - \eta \frac{\widehat{\boldsymbol{m}}_i^{(t)} + \lambda_i \boldsymbol{e}_i^{(t-1)}}{\sqrt{\widehat{\boldsymbol{v}}_i^{(t)}}+\epsilon}. \tag{9}$$

The pseudo-code of FUMO is included in Appendix A.1. Non-embedding parameters are optimized with standard Adam. When $\gamma = 1$ and $\phi(n_i) = 1$ for every item, this has the AdamD update form with bias-corrected moments. Since FUMO already applies weight decay, we recommend omitting an additional L_2 penalty on the same embeddings from $\mathcal{L}_{\text{aux}}$.

5 EXPERIMENT

5.1 EXPERIMENTAL SETUP

Datasets. We use four public sequential recommendation datasets. Yelp (Yelp) contains reviews of local businesses, LastFM (Celma, 2010) contains music listening histories, and Beauty and Sports (McAuley et al., 2015) contain Amazon product reviews. Our data preprocessing pipeline follows prior studies (Kang & McAuley, 2018; Zhou et al., 2020; 2022; Shin et al., 2024; 2025): users and items with fewer than five interactions are removed. Each user sequence's last item is held out for testing, the second-to-last for validation, and the rest for training.

Models. We implement the most representative sequential recommendation models from four different domains: (1) SASRec (Kang & McAuley, 2018) is a transformer style sequential encoder based on self-attention, (2) GRU4Rec (Hidasi et al., 2016) uses gated recurrent units (GRU) to capture sequential information, (3) FMLPRec (Zhou et al., 2022) represents frequency-domain models and replaces attention with learnable filters over MLP encoder, (4) BSARec (Shin et al., 2024) represents hybrid models that combine both self-attention and frequency-domain filters. These models do not have auxiliary loss and $\mathcal{L}_{\text{aux}} = 0$. We also include two models with auxiliary training objectives: (5) DuoRec (Qiu et al., 2022) adds a contrastive objective that counteracts representation degeneration of item embeddings. (6) SPRINT (Yang et al., 2026) is a state-of-the-art model published in 2026 which constrains attention column sums and feed-forward spectral norms to prevent the spectral collapse that amplifies popularity bias when transformers are scaled up. These two models are the representatives for two major domains: contrastive learning and regularization. Both models modify the training objective with auxiliary loss $\mathcal{L}_{\text{aux}} \neq 0$. All six models are trained with the cross-entropy recommendation loss as described in Section 3.1. Other models and methods such as Markov-Chain-based and CNN-based models are not included because they have worse performance or are not the representatives in their domain.

Optimizers. We compare six optimizers. (1) Adam (Kingma & Ba, 2015), the standard optimization choice, (2) AdamW (Loshchilov & Hutter, 2019), another standard optimization choice, (3) AdamD (Ding et al., 2025) which keeps the decay term outside both moment recurrences that the construction

FUMO's decay follows, (4) FAL (Qiu et al., 2025) which is another frequency-aware optimizer that does the opposite as FUMO by decreasing the learning rate for rare items to handle the multi-epoch overfitting problem, (5) Helen (Zhu et al., 2024), which uses frequency-dependent embedding perturbations to regularize Hessian eigenvalues, and (6) FUMO (our proposed optimizer). Note that AdamAR (Li & Lyu, 2026) is the latest work in 2026 and adapts weight decay to intervals between non-zero gradients to mitigate overfitting in sparse feature embeddings, but this mechanism is not applicable to full-category cross-entropy loss where the gradients are always non-zero.

Evaluation. We evaluate the recommendation accuracy by standard recommendation metrics HR@20 (Hit Rate) and NDCG@20 (Normalized Discounted Cumulative Gain). Following (Qiu et al., 2022; Xie et al., 2022; Shin et al., 2024; 2025), items already present in the user's sequence are removed from the ranked list.

5.2 Overall Recommendation Accuracy

Table 1 compares six optimizers across all six models on four datasets. FUMO improves on Adam in all 48 comparisons. It attains the highest reported accuracy in 39 comparisons, and the second-highest value in the remaining 9. The gains span attention-based model, RNN-based model, frequency-domain model, contrastive learning model and model that incorporate auxiliary objectives to address popularity bias. These results establish FUMO as a strong optimizer for sequential recommendation, with consistently competitive performance across all evaluated settings.

On Yelp, FUMO raises HR@20 by 7.3–10.6% and NDCG@20 by 7.8–11.5% relative to Adam across the six models. On LastFM, the corresponding gains are 10.6–18.5% and 9.6–25.0%. For example, GRU4Rec's LastFM NDCG@20 increases from 0.0320 to 0.0400, while DuoRec's HR@20 increases from 0.0844 to 0.1000. FUMO leads both metrics for every model on these two datasets. On Beauty and Sports, improvements over Adam remain positive in every comparison, although FUMO does not always lead the other optimizers.

The competing optimizers show a less consistent pattern. AdamW and AdamD do not consistently improve on Adam, suggesting that changing weight decay alone is insufficient to obtain the gains observed with FUMO. Notice that AdamD might degenerate to Adam when the optimal decay rate is small, and might degenerate to AdamW when the second moment is near-constant across items, producing same results as Adam or AdamW. FAL leads both metrics for FMLPRec on Beauty, and Helen does so for BSARec on Beauty. AdamW and FAL also attain the highest SPRINT HR@20 on Beauty and Sports, respectively. Even in these comparisons, FUMO ranks the second, and it remains among the top two optimizers across the full table.

The results of SPRINT also show that these gains persist when methods for addressing popularity bias are already present. FUMO leads both metrics with DuoRec on all four datasets and leads NDCG@20 with SPRINT throughout. This supports its use alongside contrastive learning and regularization while retaining the model and objective. In particular, the decoupled scaling in Equation (5) changes the recommendation contribution to the first moment while keeping the auxiliary contribution at its base coefficient. Frequency correction can therefore coexist with auxiliary training signals without directly multiplying them by the item-frequency factor. Taken together, the results support the combined frequency-aware update in Section 4, although aggregate accuracy alone does not isolate the contribution of each component.

5.3 Learning Progress for Rare Items

In this subsection, we examine how FUMO accelerates the learning progress of rare items. Figures 2 tracks validation accuracy during SASRec training on Yelp, Beauty, and Sports. These curves show absolute group accuracy instead of normalized progress curves in Figure 1. FUMO improves rare item accuracy early in training, reducing the delay between rapid frequent-item learning and the improvement of rare-item predictions. The advantage of FUMO at early stages is quantified at Table 2. At epoch 5, FUMO's rare item HR@20 is 0.0105 on Yelp, 0.0515 on Beauty, and 0.0230 on Sports. Compared with Adam, these correspond to relative improvements of 41.9%, 17.0%, and 66.7%, respectively. The NDCG@20 improvements are also substantial: 55.6% on Yelp, 23.5% on Beauty, and 77.5% on Sports. At epoch 10, FUMO still leads all five baselines on both metrics across all three datasets. Its HR@20 advantage over Adam narrows to 15.4% on Yelp, 7.0% on Beauty, and 9.8% on

Table 1: Overall recommendation accuracy for Adam, AdamW, AdamD, FAL, Helen, and FUMO across six models and four datasets. Bold and underlining mark the highest and second-highest distinct displayed values within each model–dataset–metric comparison.

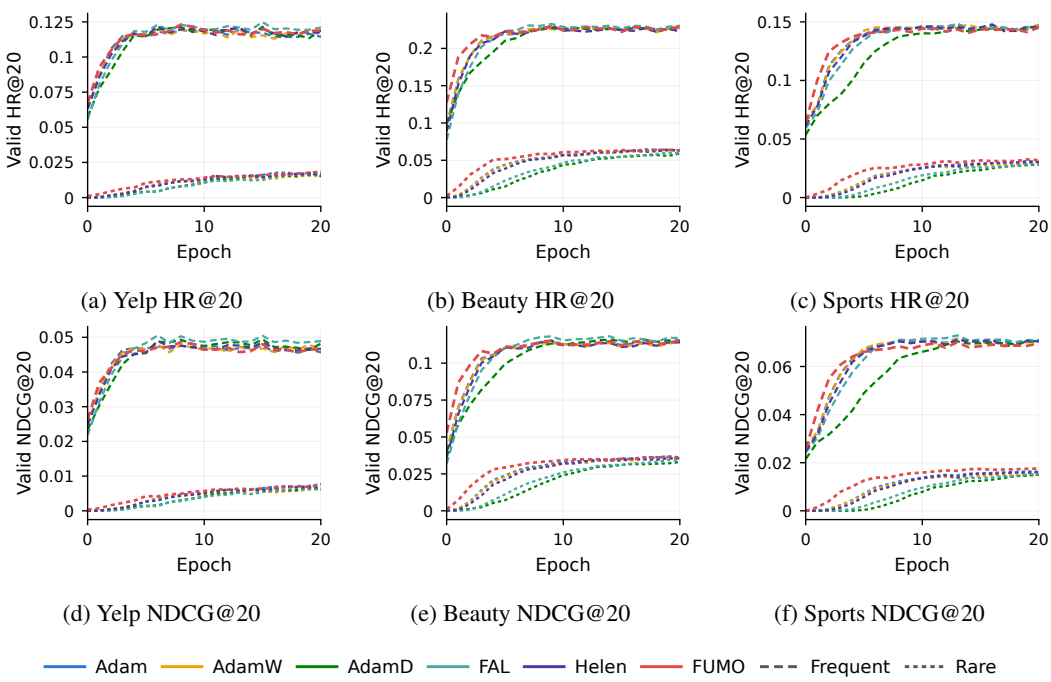
		Yelp		LastFM		Beauty		Sports	
Model	Optimizer	HR@20	NDCG@20	HR@20	NDCG@20	HR@20	NDCG@20	HR@20	NDCG@20
SASRec	Adam (Kingma & Ba, 2015)	0.0623	0.0253	0.0890	0.0452	0.1270	0.0657	0.0773	0.0377
	AdamW (Loshchilov & Hutter, 2019)	0.0619	0.0252	0.0954	0.0481	0.1275	0.0652	0.0768	0.0375
	AdamD (Ding et al., 2025)	0.0623	0.0253	0.0954	0.0481	0.1270	0.0657	0.0771	0.0379
	FAL (Qiu et al., 2025)	0.0645	0.0258	0.1028	0.0489	0.1272	0.0650	0.0780	**0.0382**
	Helen (Zhu et al., 2024)	0.0637	0.0259	0.0954	0.0485	**0.1295**	0.0661	0.0779	0.0381
	FUMO	**0.0689**	**0.0282**	**0.1037**	**0.0504**	0.1293	**0.0662**	**0.0785**	**0.0382**
GRU4Rec	Adam (Kingma & Ba, 2015)	0.0617	0.0244	0.0780	0.0320	0.0975	0.0458	0.0603	0.0265
	AdamW (Loshchilov & Hutter, 2019)	0.0630	0.0250	0.0807	0.0345	0.1013	**0.0500**	0.0614	0.0271
	AdamD (Ding et al., 2025)	0.0622	0.0250	0.0780	0.0320	0.0999	0.0464	0.0603	0.0265
	FAL (Qiu et al., 2025)	0.0612	0.0243	0.0706	0.0332	0.1006	0.0474	0.0567	0.0252
	Helen (Zhu et al., 2024)	0.0634	0.0254	0.0817	0.0366	0.1024	0.0474	0.0596	0.0260
	FUMO	**0.0662**	**0.0263**	**0.0890**	**0.0400**	**0.1052**	0.0493	**0.0638**	**0.0285**
FMLPRec	Adam (Kingma & Ba, 2015)	0.0716	0.0286	0.0972	0.0461	0.1376	0.0679	0.0853	0.0416
	AdamW (Loshchilov & Hutter, 2019)	0.0724	0.0296	0.0991	0.0444	0.1376	0.0682	0.0848	0.0413
	AdamD (Ding et al., 2025)	0.0729	0.0298	0.0972	0.0461	0.1381	0.0684	0.0853	0.0411
	FAL (Qiu et al., 2025)	0.0725	0.0294	0.0862	0.0409	**0.1400**	**0.0707**	0.0839	0.0404
	Helen (Zhu et al., 2024)	0.0732	0.0299	0.0972	0.0471	0.1358	0.0690	0.0862	0.0421
	FUMO	**0.0774**	**0.0314**	**0.1128**	**0.0541**	0.1396	0.0692	**0.0880**	**0.0426**
BSARec	Adam (Kingma & Ba, 2015)	0.0705	0.0286	0.1046	0.0492	0.1341	0.0676	0.0844	0.0412
	AdamW (Loshchilov & Hutter, 2019)	0.0721	0.0293	0.1101	0.0534	0.1346	0.0671	0.0832	0.0402
	AdamD (Ding et al., 2025)	0.0722	0.0289	0.1046	0.0495	0.1342	0.0674	0.0835	0.0403
	FAL (Qiu et al., 2025)	0.0692	0.0282	0.1000	0.0493	0.1338	0.0679	0.0841	0.0409
	Helen (Zhu et al., 2024)	0.0718	0.0292	0.1064	0.0519	**0.1382**	**0.0693**	0.0853	**0.0419**
	FUMO	**0.0779**	**0.0316**	**0.1165**	**0.0549**	0.1369	0.0691	**0.0874**	0.0416
DuoRec	Adam (Kingma & Ba, 2015)	0.0708	0.0291	0.0844	0.0427	0.1300	0.0663	0.0783	0.0381
	AdamW (Loshchilov & Hutter, 2019)	0.0740	0.0304	0.0908	0.0448	0.1291	0.0659	0.0783	0.0385
	AdamD (Ding et al., 2025)	0.0690	0.0287	0.0844	0.0427	0.1300	0.0663	0.0788	0.0386
	FAL (Qiu et al., 2025)	0.0718	0.0297	0.0789	0.0384	0.1293	0.0651	0.0768	0.0371
	Helen (Zhu et al., 2024)	0.0710	0.0294	0.0927	0.0452	0.1304	0.0667	0.0783	0.0387
	FUMO	**0.0760**	**0.0316**	**0.1000**	**0.0501**	**0.1313**	**0.0671**	**0.0792**	**0.0388**
SPRINT	Adam (Kingma & Ba, 2015)	0.0627	0.0253	0.0954	0.0457	0.1292	0.0660	0.0762	0.0376
	AdamW (Loshchilov & Hutter, 2019)	0.0624	0.0256	0.0890	0.0451	**0.1306**	0.0663	0.0773	0.0374
	AdamD (Ding et al., 2025)	0.0633	0.0260	0.0917	0.0447	0.1298	0.0662	0.0760	0.0374
	FAL (Qiu et al., 2025)	0.0637	0.0264	0.0881	0.0439	0.1294	0.0649	**0.0789**	0.0386
	Helen (Zhu et al., 2024)	0.0633	0.0262	0.1018	0.0477	0.1278	0.0661	0.0781	0.0386
	FUMO	**0.0676**	**0.0280**	**0.1055**	**0.0501**	0.1303	**0.0664**	0.0787	**0.0393**

Table 2: Rare-item validation accuracy for SASRec at epochs 5 and 10. Bold and underlining mark the highest and second-highest distinct displayed values within each dataset, epoch, and metric.

		Yelp		Beauty		Sports	
Epoch	Optimizer	HR@20	NDCG@20	HR@20	NDCG@20	HR@20	NDCG@20
5	Adam (Kingma & Ba, 2015)	0.0074	0.0027	0.0440	0.0238	0.0138	0.0071
	AdamW (Loshchilov & Hutter, 2019)	0.0071	0.0026	0.0431	0.0230	0.0138	0.0067
	AdamD (Ding et al., 2025)	0.0042	0.0014	0.0148	0.0072	0.0012	0.0003
	FAL (Qiu et al., 2025)	0.0041	0.0013	0.0227	0.0111	0.0049	0.0019
	Helen (Zhu et al., 2024)	0.0078	0.0026	0.0386	0.0210	0.0118	0.0057
	FUMO	**0.0105**	**0.0042**	**0.0515**	**0.0294**	**0.0230**	**0.0126**
10	Adam (Kingma & Ba, 2015)	0.0123	0.0051	0.0570	0.0330	0.0256	0.0139
	AdamW (Loshchilov & Hutter, 2019)	0.0107	0.0045	0.0569	0.0318	0.0250	0.0135
	AdamD (Ding et al., 2025)	0.0105	0.0041	0.0440	0.0241	0.0147	0.0079
	FAL (Qiu et al., 2025)	0.0106	0.0040	0.0467	0.0260	0.0189	0.0096
	Helen (Zhu et al., 2024)	0.0131	0.0050	0.0570	0.0321	0.0253	0.0136
	FUMO	**0.0142**	**0.0057**	**0.0610**	**0.0343**	**0.0281**	**0.0159**

Sports as the baselines make further progress. Higher HR@20 means that more rare targets enter the top-20 recommendation list, while higher NDCG@20 indicates an improvement in their rank quality. These results show that the early benefit extends beyond a reduction in training loss.

This result is consistent with the update design in Section 4.1, which assigns larger relative recommendation steps to rarer embedding rows. On the other hand, FAL decreases the learning rate for rare items and remains below Adam on every rare item metric at epochs 5 and 10.



(a) Yelp HR@20 (b) Beauty HR@20 (c) Sports HR@20

(d) Yelp NDCG@20 (e) Beauty NDCG@20 (f) Sports NDCG@20

Figure 2: Validation HR@20 and NDCG@20 during SASRec training. Dashed lines are frequent items and dotted lines are rare items. FUMO raises rare-item hit rate earlier than the five baseline optimizers; the advantage persists the longest on Sports.

Summary: Across all models and datasets, FUMO demonstrates robust improvement on both HR@20 up to 18.5% and NDCG@20 up to 25.0% over Adam. The trajectories also show better rare item performance at early stages. These findings support allocating larger relative recommendation steps to rare items within a shared training schedule, with the benefit depending on the model and dataset. Although the magnitude of performance gain varies across datasets, the combination of consistent overall gains, earlier rare item learning, and compatibility with diverse models makes FUMO a competitive optimizer for sequential recommendation.

6 CONCLUSION

We introduced FUMO to address unequal learning progress across item frequencies in sequential recommendation. FUMO combines frequency-aware step sizes and weight decay adapted to frequency and gradient moments without changing the model architecture or adding a loss term. Experiments across six models and four datasets demonstrate the strength and consistency of this approach. FUMO shows superior performance over Adam in all models and datasets with improvements up to 18.5% in HR@20 and 25.0% in NDCG@20 and achieves the highest or second-highest accuracy over other baseline optimizers. Training trajectories also show earlier improvements in rare item recommendation accuracy. Together, these results make FUMO a compelling optimizer for sequential recommendation. These results suggest that item frequency is a useful signal for adjusting embedding optimization across different sequential recommendation models.

We identify two potential future directions for FUMO. First, the frequency multipliers are fixed before training; adapting them using recent interaction counts and observed learning progress could help the optimizer respond to changing item popularity and reduce acceleration once rare items are well learned. Second, further theoretical developments could establish convergence and generalization guarantees for frequency-aware optimization.

AI USE STATEMENT

We used generative AI tools to aid and polish writing, including improving clarity, conciseness, and organization, sketching the proof in Appendix A.2 , and for retrieval and discovery, including identifying relevant literature on sequential recommendation, popularity bias, and frequency-aware optimization. We have not used generative AI tools to develop theoretical framework, design research methodology, propose hypothesis, or implement the methods. Generating synthetic datasets, qualitative or thematic data analysis, and dataset cleaning are not applicable to this work. We have reviewed all AI-assisted work. Specifically, we verified the correctness of proof sketch and checked every reference identified with AI assistance against the original publication before it was cited. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

All authors have read and adhere to the ICLR Code of Ethics. This work involves no human-subject experiments, user studies, crowdsourcing, or new data collection. We use four public benchmark datasets (Yelp, LastFM, Amazon Beauty, and Amazon Sports) that are widely used in sequential recommendation research, under the terms of their original releases. Training uses only each user's sequence of item IDs; we use no names, review text, ratings, or other personal attributes, and we make no attempt to identify users.

REPRODUCIBILITY STATEMENT

Section 4 specifies the complete FUMO update, and Algorithm 1 in Appendix A.1 gives its pseudo-code. All four datasets are public. Appendix B.2 lists the codebases we build on, how each baseline optimizer is implemented, the shared training settings, the hyperparameter search spaces, the early-stopping rule, the evaluation protocol, and the software and hardware environment. It also documents one correction that matters when comparing with published numbers: the official BSARec code does not fully exclude previously interacted items from the ranking, so we fix this for all methods, and our numbers differ from those reported in those papers. We will release the code publicly upon publication.

REFERENCES

Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. Managing popularity bias in recommender systems with personalized re-ranking. In *Proceedings of the Thirty-Second International Florida Artificial Intelligence Research Society Conference*, pp. 413–418. AAAI Press, 2019. URL https://arxiv.org/abs/1901.07555.

Shaden Alshammari, Yu-Xiong Wang, Deva Ramanan, and Shu Kong. Long-tailed recognition via weight balancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. URL https://arxiv.org/abs/2203.14197.

Miaomiao Cai, Lei Chen, Yifan Wang, Haoyue Bai, Peijie Sun, Le Wu, Min Zhang, and Meng Wang. Popularity-aware alignment and contrast for mitigating popularity bias. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2024. doi: 10.1145/3637528.3671824. URL https://arxiv.org/abs/2405.20718.

Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems 32*, 2019. URL https://arxiv.org/abs/1906.07413.

Òscar Celma. *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer, 2010. doi: 10.1007/978-3-642-13287-2.

Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. In *Advances in Neural Information Processing Systems 33*, pp. 8765–8775, 2020. URL https://arxiv.org/abs/2007.00224.

Kuangyu Ding, Nachuan Xiao, and Kim-Chuan Toh. Adam-family methods with decoupled weight decay in deep learning. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=xVEHiAZ7uR.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011. URL https://jmlr.org/papers/v12/duchi11a.html.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=6Tm1mposlrM.

Corrado Gini. Measurement of inequality of incomes. *The Economic Journal*, 31(121):124–125, 1921. doi: 10.2307/2223319.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 4th edition, 2013.

Priyanka Gupta, Diksha Garg, Pankaj Malhotra, Lovekesh Vig, and Gautam Shroff. NISER: Normalized item and session representations to handle popularity bias, 2019. URL https://arxiv.org/abs/1909.04276.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1511.06939.

Wang-Cheng Kang and Julian J. McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining*, pp. 197–206, 2018. doi: 10.1109/ICDM.2018.00035.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations*, 2015. doi: 10.48550/arXiv.1412.6980. URL https://arxiv.org/abs/1412.6980.

Anastasiia Klimashevskaia, Dietmar Jannach, Mehdi Elahi, and Christoph Trattner. A survey on popularity bias in recommender systems. *User Modeling and User-Adapted Interaction*, 34(5): 1777–1834, 2024. doi: 10.1007/s11257-024-09406-0. URL https://arxiv.org/abs/2308.01118.

Frederik Kunstner, Robin Yadav, Alan Milligan, Mark Schmidt, and Alberto Bietti. Heavy-tailed class imbalance and why Adam outperforms gradient descent on language models. In *Advances in Neural Information Processing Systems 37*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/350e718ff74062b4bac2c6ffd9e1ac66-Abstract-Conference.html.

Mang Li and Wei Lyu. Adaptive regularization for large-scale sparse feature embedding models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=QFH5mwP9oH.

Yan Li, Dhruv Choudhary, Xiaohan Wei, Baichuan Yuan, Bhargav Bhushanam, Tuo Zhao, and Guanghui Lan. Frequency-aware SGD for efficient embedding learning with provable benefits. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=ibqTBNfJmi.

Siyi Lin, Chongming Gao, Jiawei Chen, Sheng Zhou, Binbin Hu, Yan Feng, Chun Chen, and Can Wang. How do recommendation models amplify popularity bias? an analysis from the spectral perspective. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, pp. 659–668. Association for Computing Machinery, 2025. doi: 10.1145/3701551.3703579. URL https://arxiv.org/abs/2404.12008.

Siyi Liu and Yujia Zheng. Long-tail session-based recommendation. In *Fourteenth ACM Conference on Recommender Systems*, pp. 509–514. Association for Computing Machinery, 2020. doi: 10.1145/3383313.3412222.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations*, 2019. doi: 10.48550/arXiv.1711.05101. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Attila Máté. On the equality of mixed partial derivatives, September 2011. URL https://www.math.purdue.edu/~eremenko/dvi/mixedpartial.pdf. Course notes, Mathematics 2201, Brooklyn College, City University of New York.

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*, pp. 43–52. Association for Computing Machinery, 2015. doi: 10.1145/2766462.2767755.

Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=37nvvqkCo5.

Andrew Qiu, Runze Su, Ling Leng, Shubham Barhate, Rafael Rios Müller, Han Sun, Zhifang Liu, Hin Wai Lui, Kungang Li, and Shayan Ehsani. The evolution of embedding table optimization and multi-epoch training in Pinterest ads conversion. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2025. URL https://arxiv.org/abs/2505.05605.

Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. Contrastive learning for representation degeneration problem in sequential recommendation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pp. 813–823. Association for Computing Machinery, 2022. doi: 10.1145/3488560.3498433.

Weijieying Ren, Lei Wang, Kunpeng Liu, Ruocheng Guo, Ee-Peng Lim, and Yanjie Fu. Mitigating popularity bias in recommendation with unbalanced interactions: A gradient perspective. In *2022 IEEE International Conference on Data Mining (ICDM)*, pp. 438–447. IEEE, 2022. doi: 10.1109/ICDM54844.2022.00054. URL https://arxiv.org/abs/2211.01154.

Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, pp. 811–820. Association for Computing Machinery, 2010. doi: 10.1145/1772690.1772773.

Yuta Saito, Suguru Yaginuma, Yuta Nishino, Hayato Sakata, and Kazuhide Nakata. Unbiased recommender learning from missing-not-at-random implicit feedback. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. Association for Computing Machinery, 2020. doi: 10.1145/3336191.3371783. URL https://arxiv.org/abs/1909.03601.

Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. Recommendations as treatments: Debiasing learning and evaluation. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1670–1679. PMLR, 2016. URL https://proceedings.mlr.press/v48/schnabel16.html.

Yehjin Shin, Jeongwhan Choi, Hyowon Wi, and Noseong Park. An attentive inductive bias for sequential recommendation beyond the self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 8984–8992, 2024. doi: 10.1609/aaai.v38i8.28747.

Yehjin Shin, Jeongwhan Choi, Seojin Kim, and Noseong Park. TV-rec: Time-variant convolutional filter for sequential recommendation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=cWEssTIwG5.

Harald Steck. Item popularity and recommendation accuracy. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, pp. 125–132. Association for Computing Machinery, 2011. doi: 10.1145/2043932.2043957.

Harald Steck. Calibrated recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pp. 154–162. Association for Computing Machinery, 2018. doi: 10.1145/3240323.3240372.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 1441–1450. Association for Computing Machinery, 2019. doi: 10.1145/3357384.3357895.

Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 565–573. Association for Computing Machinery, 2018. doi: 10.1145/3159652.3159656.

Tianxin Wei, Fuli Feng, Jiawei Chen, Ziwei Wu, Jinfeng Yi, and Xiangnan He. Model-agnostic counterfactual reasoning for eliminating popularity bias in recommender system. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1791–1800. Association for Computing Machinery, 2021. doi: 10.1145/3447548.3467289.

Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. Contrastive learning for sequential recommendation. In *2022 IEEE 38th International Conference on Data Engineering*, pp. 1259–1273. IEEE, 2022. doi: 10.1109/ICDE53745.2022.00099.

Weiqin Yang, Yue Pan, Chongming Gao, Sheng Zhou, Xiang Wang, Can Wang, and Jiawei Chen. The pitfall of scaling up: Uncovering and mitigating popularity bias amplification in scaling transformer-based recommenders. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2026. doi: 10.1145/3770855.3818185. URL https://arxiv.org/abs/2606.21911.

Yelp. Yelp open dataset. URL https://www.yelp.com/dataset.

Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Nguyen Quoc Viet Hung. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1294–1303. Association for Computing Machinery, 2022. doi: 10.1145/3477495.3531937. URL https://arxiv.org/abs/2112.08679.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://arxiv.org/abs/2001.06782.

Yin Zhang, Ruoxi Wang, Tiansheng Yao, Xinyang Yi, Lichan Hong, James Caverlee, Ed H. Chi, and Derek Zhiyuan Cheng. Empowering long-tail item recommendation through cross decoupling network (CDN). In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2023. doi: 10.1145/3580305.3599814.

Zihao Zhao, Jiawei Chen, Sheng Zhou, Xiangnan He, Xuezhi Cao, Fuzheng Zhang, and Wei Wu. Popularity bias is not always evil: Disentangling benign and harmful bias for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 35(10):9920–9931, 2023. doi: 10.1109/TKDE.2022.3218994.

Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Depeng Jin, and Yong Li. Disentangling user interest and conformity for recommendation with causal embedding. In *Proceedings of the Web Conference 2021*. Association for Computing Machinery, 2021. doi: 10.1145/3442381.3449788. URL https://arxiv.org/abs/2006.11011.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S^3-Rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, pp. 1893–1902. Association for Computing Machinery, 2020. doi: 10.1145/3340531.3411954.

Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. Filter-enhanced MLP is all you need for sequential recommendation. In *Proceedings of the ACM Web Conference 2022*, pp. 2388–2399. Association for Computing Machinery, 2022. doi: 10.1145/3485447.3512111.

Zirui Zhu, Yong Liu, Zangwei Zheng, Huifeng Guo, and Yang You. Helen: Optimizing CTR prediction models with frequency-wise hessian eigenvalue regularization. In *Proceedings of the ACM on Web Conference 2024*, pp. 3485–3496. Association for Computing Machinery, 2024. doi: 10.1145/3589334.3645463.

APPENDIX CONTENTS

A FUMO Pseudo Code and Analysis **15**

A.1 Pseudo Code for FUMO 15

A.2 Can a Scalar Loss Reproduce the Scaled Gradient? 15

A.3 Comparison with Existing Frequency-Aware Optimizers 17

B Experiment Details **18**

B.1 Dataset Statistics 18

B.2 Implementation and Details 18

B.3 Empirical Convergence Behavior 19

B.4 Effect of FUMO on Popularity Bias 23

B.5 Ablation Study 25

B.6 Hyperparameter Sensitivity 26

A FUMO PSEUDO CODE AND ANALYSIS

A.1 PSEUDO CODE FOR FUMO

Algorithm 1 summarizes the procedure. We precompute ρ_i and λ_i, then update and bias-correct the two moment buffers at each iteration before applying shrinkage and the loss step.

Algorithm 1 FUMO embedding update

Require: Initial embedding layer $E^{(0)}$; item counts $\{n_i\}_{i=1}^{M}$; number of iterations T
Hyperparameters: $r, \gamma, \eta, \lambda, \beta_1, \beta_2, \epsilon$

for each item i **do**
  $\rho_i \leftarrow \gamma\phi_r(n_i)$, $\lambda_i \leftarrow \lambda\phi_r(n_i)$
  $\boldsymbol{m}_i^{(0)} \leftarrow \mathbf{0}$, $\boldsymbol{v}_i^{(0)} \leftarrow \mathbf{0}$
for $t = 1, \ldots, T$ **do**
  Evaluate all gradients at the current parameters
  for each item i **do**
    $\boldsymbol{g}_{\mathrm{rec},i}^{(t)} \leftarrow \nabla_{\boldsymbol{e}_i}\mathcal{L}_{\mathrm{rec}}$, $\boldsymbol{g}_{\mathrm{aux},i}^{(t)} \leftarrow \nabla_{\boldsymbol{e}_i}\mathcal{L}_{\mathrm{aux}}$
    $\boldsymbol{g}_i^{(t)} \leftarrow \boldsymbol{g}_{\mathrm{rec},i}^{(t)} + \boldsymbol{g}_{\mathrm{aux},i}^{(t)}$
    $\boldsymbol{m}_i^{(t)} \leftarrow \beta_1 \boldsymbol{m}_i^{(t-1)} + (1-\beta_1)(\rho_i \boldsymbol{g}_{\mathrm{rec},i}^{(t)} + \boldsymbol{g}_{\mathrm{aux},i}^{(t)})$
    $\boldsymbol{v}_i^{(t)} \leftarrow \beta_2 \boldsymbol{v}_i^{(t-1)} + (1-\beta_2)(\boldsymbol{g}_i^{(t)} \odot \boldsymbol{g}_i^{(t)})$
    $\widehat{\boldsymbol{m}}_i^{(t)} \leftarrow \boldsymbol{m}_i^{(t)}/(1-\beta_1^t)$, $\widehat{\boldsymbol{v}}_i^{(t)} \leftarrow \boldsymbol{v}_i^{(t)}/(1-\beta_2^t)$
    $\boldsymbol{s}_i^{(t)} \leftarrow \min\{\eta\lambda_i/(\sqrt{\widehat{\boldsymbol{v}}_i^{(t)}} + \epsilon), \mathbf{1}\}$
    $\boldsymbol{e}_i^{(t)} \leftarrow (\mathbf{1} - \boldsymbol{s}_i^{(t)}) \odot \boldsymbol{e}_i^{(t-1)} - \eta \dfrac{\widehat{\boldsymbol{m}}_i^{(t)}}{\sqrt{\widehat{\boldsymbol{v}}_i^{(t)}} + \epsilon}$

A.2 CAN A SCALAR LOSS REPRODUCE THE SCALED GRADIENT?

Could we replace the loss by a scalar function G whose gradient already contains FUMO's recommendation multipliers? In general, scaling coupled embedding rows differently prevents such a function from existing. The obstruction is exact: mixed derivatives of a scalar loss must agree, whereas unequal row multipliers can destroy this agreement.

Fix the encoder parameters θ, a nonempty batch $\mathcal{B}$ of training examples $(\boldsymbol{x}_u, y_u)$, and any stochastic choices in the forward pass. Here $\mathcal{L}_{\mathrm{rec}}$ denotes the mean recommendation loss on $\mathcal{B}$, and $\mathcal{L}_{\mathrm{aux}}$ denotes the corresponding auxiliary loss. Let $\Omega \subset \mathbb{R}^{M\times d}$ be an open region on which both losses are twice continuously differentiable. The multipliers ρ_i are fixed by Equation (4). Define the vector field with row blocks

$$\boldsymbol{F}_i(E) = \rho_i \nabla_{\boldsymbol{e}_i} \mathcal{L}_{\mathrm{rec}}(E) + \nabla_{\boldsymbol{e}_i} \mathcal{L}_{\mathrm{aux}}(E). \tag{10}$$

This is the gradient combination entering the first moment in Equation (5). We ask whether one differentiable function $G : \Omega \to \mathbb{R}$ can satisfy $\nabla_{\boldsymbol{e}_i} G(E) = \boldsymbol{F}_i(E)$ for every row i and every $E \in \Omega$. Matching the field throughout a region is stronger than matching it at a single iterate.

Proposition A.1 (Obstruction to an equivalent scalar loss)**.** *For rows i, j, let $H_{ij}(E) = \partial(\nabla_{\boldsymbol{e}_i}\mathcal{L}_{\mathrm{rec}})/\partial \boldsymbol{e}_j^\top \in \mathbb{R}^{d\times d}$ be the corresponding Hessian block. A necessary condition for such a G to exist is*

$$(\rho_i - \rho_j) H_{ij}(E) = 0 \qquad \textit{for all } i, j \textit{ and } E \in \Omega. \tag{11}$$

Consequently, if $\rho_i \neq \rho_j$ and $H_{ij}(E_0) \neq 0$ for some $E_0 \in \Omega$, no such scalar loss exists, even on a neighborhood of E_0.

Proof. Because $\boldsymbol{F}$ is continuously differentiable, any G with $\nabla G = \boldsymbol{F}$ is twice continuously differentiable. Its Hessian must therefore be symmetric by equality of mixed partial derivatives (Máté, 2011, Theorem 1). The same symmetry holds for the Hessians of both losses. Thus,

$$\frac{\partial \boldsymbol{F}_i}{\partial \boldsymbol{e}_j^\top} - \left(\frac{\partial \boldsymbol{F}_j}{\partial \boldsymbol{e}_i^\top}\right)^\top = (\rho_i - \rho_j) H_{ij}. \tag{12}$$

The auxiliary Hessian cancels because its row coefficients are all one. The left-hand side must vanish if $\boldsymbol{F} = \nabla G$, proving the claim. □

The full softmax supplies the coupling. Consider two items $i \neq j$ with different counts $n_i \neq n_j$ that occur in none of the input sequences in $\mathcal{B}$. Assume the encoder accesses the item table only through the embeddings of the input items. Then each $\boldsymbol{h}_u$ is independent of $\boldsymbol{e}_i$ and $\boldsymbol{e}_j$, even though the table is tied. The input contribution in Equation (3) vanishes for these rows. Differentiating the remaining output contribution using Equation (1) gives

$$H_{ij}(E) = -\frac{1}{|\mathcal{B}|} \sum_{u\in\mathcal{B}} p_{u,i} p_{u,j} \boldsymbol{h}_u \boldsymbol{h}_u^\top. \tag{13}$$

Here the sum runs over the examples in the batch, and $|\mathcal{B}|$ is its size. At finite logits, $p_{u,i}p_{u,j} > 0$. If at least one $\boldsymbol{h}_u \neq \boldsymbol{0}$ at E_0, then

$$\operatorname{tr} H_{ij}(E_0) = -\frac{1}{|\mathcal{B}|} \sum_{u\in\mathcal{B}} p_{u,i} p_{u,j} \|\boldsymbol{h}_u\|_2^2 < 0, \tag{14}$$

where tr denotes the matrix trace and $\|\cdot\|_2$ the Euclidean norm. Hence $H_{ij}(E_0) \neq 0$. Since ϕ_r is strictly decreasing, $n_i \neq n_j$ implies $\rho_i \neq \rho_j$, so Proposition A.1 rules out G. The items may be prediction targets; only their absence from the input sequences is needed. A joint loss $G(E, \theta)$ cannot avoid this obstruction, because fixing θ would yield the scalar function already ruled out above.

Why an equivalent scalar loss is highly unlikely in real-world datasets. The absent-item argument gives a direct certificate of nonexistence, but the underlying obstruction applies to every pair of rows. By Equation (11), existence of G requires the recommendation cross-Hessian to vanish between *every* pair of items with different counts, at *every* embedding configuration in the region. Full softmax couples these items through its shared normalizer. When items also occur in the input sequences, encoder dependence adds further Hessian terms. For G to exist, these terms would have to cancel the output coupling exactly, simultaneously across all different-frequency pairs and throughout the region. Neither the tied embedding architecture nor the cross-entropy objective imposes these count-dependent cancellation identities. This makes an equivalent scalar loss highly unlikely in a general tied-embedding recommender: its existence would require a special structure that eliminates all coupling between differently scaled rows. For example, a loss separable into groups of rows sharing a multiplier has this structure, whereas the full-softmax calculation above explicitly violates

it under the stated conditions. The unchanged auxiliary gradient cannot restore the missing symmetry, since its Hessian cancels in the proof.

The optimizer adds a further distinction. Even if an exceptional loss admits such a G, ordinary Adam applied to G generally changes the second moment, while FUMO retains the unscaled total gradient in Equation (6). Reproducing the complete update therefore requires more than an equivalent scalar loss for the first moment.

A.3 Comparison with Existing Frequency-Aware Optimizers

FUMO addresses a specific optimization problem: rare items must make useful recommendation progress before training is stopped by the aggregate validation criterion. Frequency-aware optimization already provides several ways to treat embedding rows differently. This section compares how FA-SGD, FAL, Helen, and FUMO use frequency information.

Beyond compensating for sparse updates. FA-SGD (Li et al., 2022) is the closest precedent for giving rare embeddings larger steps. Its analysis establishes frequency-dependent convergence benefits for sparse embedding learning (Li et al., 2022, Theorem 3.1). In that formulation, each sampled pair updates only its associated embedding rows. Our tied item table has a different structure: the full-softmax output term in Equation (3) generally contributes a gradient even when an item is absent from the input and is not the target. Receiving a nonzero gradient therefore does not mean that an item receives equally informative training. FUMO uses frequency to adjust recommendation progress in this dense-update setting, together with Adam-style moments and explicit decay. The contribution extends beyond the shared principle of assigning larger steps to rare embeddings; FA-SGD's sparse-update guarantees do not directly establish convergence for this update.

Accelerating learning while controlling overfitting. FAL (Qiu et al., 2025) scales embedding learning rates by relative log frequency, assigning smaller rates to rarer rows to mitigate multi-epoch overfitting in ads conversion models. FUMO responds to the delayed rare-item progress observed in Section 3.2 by increasing their relative recommendation steps. It addresses the associated overfitting risk through a separate decay term. Specifically, the recommendation multiplier $\rho_i = \gamma\phi_r(n_i)$ and decay coefficient $\lambda_i = \lambda\phi_r(n_i)$ share the decreasing frequency profile from Section 4.1, while γ and λ control their strengths separately. Equation (7) also adapts shrinkage to the second moment and clips it to prevent decay alone from reversing a coordinate's sign. This design permits stronger recommendation updates for rare items while constraining their embedding magnitudes. Its motivation is to support continued learning without using a smaller rare-item learning rate as the means of regularization.

Controlling updates rather than perturbation radii. Helen (Zhu et al., 2024) relates feature frequency to the largest embedding Hessian eigenvalues in click-through rate prediction. It uses sharpness-aware minimization with stronger perturbations for frequent features, subject to a lower bound, and evaluates the update gradient at the perturbed parameters. FUMO directly modifies the recommendation contribution and shrinkage at the current parameters. It therefore does not require Helen's additional gradient evaluation at a perturbed model.

A distinct treatment of loss contributions and moments. The update in Equations (5)–(6) makes the difference concrete. FUMO multiplies only the recommendation gradient entering the first moment and computes the second moment from the unscaled total gradient. Scaling the entire embedding step would also multiply the auxiliary contribution. Feeding a gradient scaled by a fixed row multiplier into both Adam moments would allow the normalization to largely cancel that scaling when the numerical stabilizer is negligible. FUMO avoids both effects with one first-moment buffer and one second-moment buffer per row. Auxiliary gradients retain coefficient one in the numerator, although they still interact with recommendation gradients through the shared denominator and the training trajectory. When auxiliary gradients are absent and decay is disabled, the fixed multiplier can be factored out of the first-moment recurrence, recovering row-scaled Adam. With auxiliary losses and decay present, the selective scaling and shrinkage specify additional behavior that a single row learning rate cannot reproduce in general.

The contribution relative to these methods is thus a coordinated update for tied sequential recommenders: it accelerates the recommendation component of rare-item learning, retains auxiliary contributions at their base numerator scale, and controls embedding growth through frequency- and second-moment-dependent decay. Each component has a distinct role in addressing unequal learning progress. The component ablations in Appendix B.5 assess their empirical contributions.

B Experiment Details

B.1 Dataset Statistics

Table 3 summarizes the four datasets used in Section 5.1. All items have at least five interactions before the data are split.

We define the frequent group that contains the $\lceil 0.2M \rceil$ items with the largest full dataset frequencies, and the rare group contains all remaining items. Ties at the boundary may be broken arbitrarily without changing the reported statistics. Group averages are the total occurrences in each group divided by its number of items. The Gini coefficient (Gini, 1921) is

$$G = \frac{\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} |n_i - n_j|}{2M \sum_{i \in \mathcal{I}} n_i}.$$

Larger values indicate greater inequality in item frequency.

Table 3: Dataset statistics based on the entire filtered dataset. Rare and frequent groups contain the bottom 80% and top 20% of the items.

			Item frequency			Number of items		Interactions			Avg. frequency		
Dataset	Users	Items	Min.	Max.	Avg.	Rare	Freq.	Total	Rare (%)	Freq. (%)	Rare	Freq.	Gini
Yelp	30,431	20,033	5	500	15.79	16,026	4,007	316,354	46.49	53.51	9.18	42.25	0.457
LastFM	1,090	3,646	5	177	14.41	2,916	730	52,551	47.70	52.30	8.60	37.65	0.437
Beauty	22,363	12,101	5	431	16.40	9,680	2,421	198,502	41.89	58.11	8.59	47.64	0.499
Sports	35,598	18,357	5	1,042	16.14	14,685	3,672	296,337	43.17	56.83	8.71	45.86	0.490

B.2 Implementation and Details

Implementation. Our training framework builds on the official BSARec codebase (Shin et al., 2024),[1] from which we inherit the implementations of SASRec, GRU4Rec, FMLPRec, DuoRec, and BSARec. We implement SPRINT (Yang et al., 2026) using its official codebase[2] and use the same SASRec implementation as the sequence encoder for both DuoRec and SPRINT. In our SPRINT implementation, spectral norm regularization is not applied to the embedding layer. We implement AdamD (Ding et al., 2025) and FAL (Qiu et al., 2025) from the pseudocode in their respective papers, and adapt Helen (Zhu et al., 2024) from its official implementation.[3]

Shared training settings. Across all experiments, we fix the base learning rate at $\eta = 10^{-3}$, the batch size at 256, and the hidden dimension at 64. Following BSARec (Shin et al., 2024), we set the maximum sequence length to 50. We use Adam's default first- and second-moment coefficients, $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and do not apply gradient clipping. For AdamD and FUMO, the numerical stabilizer is fixed at $\epsilon = 10^{-8}$. Weight decay is applied only to the embedding layer in FUMO and to all parameters in the other optimizers.

Training and evaluation. Following the BSARec training setup (Shin et al., 2024), we train each model–optimizer combination for up to 200 epochs. The experiments are validated by NDCG@20. We apply early-stopping when the validation score does not increase for 10 consecutive epochs. Following the full-catalog evaluation protocol used by DuoRec and BSARec (Qiu et al., 2022; Shin

[1] BSARec: `https://github.com/yehjin-shin/BSARec`.

[2] SPRINT is provided in GenRec: `https://github.com/Tiny-Snow/GenRec`.

[3] Helen: `https://github.com/NUS-HPC-AI-Lab/Helen`.

et al., 2024), we rank candidate items over the entire catalog after excluding items in the user's observed interaction history. We implement this exclusion by setting the logits of previously interacted items to $-\infty$ before ranking[4]. We report Hit Rate (HR@20) and Normalized Discounted Cumulative Gain (NDCG@20).

Let r_u be the rank of user u's held-out target item y_u after excluding previously interacted items, with rank 1 being the highest. The metrics are computed by

$$\begin{aligned} \text{HR@20} &= \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbb{1}\{r_u \le 20\}, \\ \text{NDCG@20} &= \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{\mathbb{1}\{r_u \le 20\}}{\log_2(r_u + 1)}. \end{aligned} \tag{15}$$

HR@20 measures how often the target appears in the top 20, while NDCG@20 additionally rewards higher ranks of the target item.

Hyperparameter selection. We apply grid search on optimizer hyperparameters. For FUMO, we tune the global step size multiplier γ, the frequency-scale parameter r, and the base weight-decay coefficient λ over

$$\begin{aligned} &\gamma \in \{1, 2, 3, 4, 5\}, \\ &r \in \{1, 5, 10, 20\}, \\ &\lambda \in \{5 \times 10^{-5}, 1 \times 10^{-5}, 5 \times 10^{-6}, 1 \times 10^{-6}\}. \end{aligned}$$

AdamW is tuned with $\lambda = \{0.1, 0.01\}$ FAL has no tunable hyperparameter. We tune Helen following it official instructions. We restrict the weight decay search for AdamD to $\lambda \in \{1 \times 10^{-5}, 5 \times 10^{-6}, 1 \times 10^{-6}\}$ so that every candidate satisfies $\eta \le \epsilon/\lambda$ (Ding et al., 2025) under the fixed learning rate and numerical stabilizer. AdamD and FUMO uses smaller λ than AdamW because the division of second moment makes the shrinkage numerically larger.

To match the hyperparameter tuning budgets with other baseline optimizers, we also compare the results of FUMO from a restricted tuning budget. Specifically, we compare the results within these four configurations of (γ, r, λ)

- small γ, moderate r, large λ : $(1, 5, 5 \times 10^{-5})$
- moderate γ, moderate r, large λ : $(3, 5, 5 \times 10^{-5})$
- large γ, moderate r, moderate λ : $(5, 5, 5 \times 10^{-6})$
- large γ, another moderate r, moderate λ : $(5, 10, 5 \times 10^{-6})$

This reduces the number of configurations from 80 to 4, a 95% reduction. Table 4 shows that FUMO exceeds the best baseline result in 31 of 48 comparisons and one tie. FUMO also achieves higher accuracy compared to Adam in 41 comparisons and ties in one. This result shows that FUMO still preserves good performance with restricted tuning budgets.

Computing environment. All experiments run on a Linux server equipped with an NVIDIA H800 GPU, using a virtual environment with Python 3.10 and PyTorch 2.11.

B.3 Empirical Convergence Behavior

Figure 3 compares training loss for frequent and rare targets. Frequent-item loss falls quickly across the plotted optimizers, whereas FUMO reduces rare-item loss earlier than the baselines. This agrees with the earlier improvement in rare-item validation accuracy in Section 5.3. AdamD eventually reaches lower training loss without consistently achieving higher validation accuracy, illustrating that additional loss reduction need not translate into better recommendation quality.

Figure 4 compares the embedding-gradient norms of the overall training cross entropy, grouped by item frequency. Both rare and frequent groups show the same trend. FUMO avoids the pronounced

[4]The released official implementations of BSARec (Shin et al., 2024) and TV-Rec (Shin et al., 2025) score interacted items 0, which does not remove them because logits can be negative. Here, we fix this issue by assigning $-\infty$ instead. This results in a different evaluation results compared to their reported values.

Table 4: Recommendation accuracy on restricted tuning budget. Bold and underlining mark the highest and second-highest distinct displayed values.

		Yelp		LastFM		Beauty		Sports	
Model	Optimizer	HR@20	NDCG@20	HR@20	NDCG@20	HR@20	NDCG@20	HR@20	NDCG@20
SASRec	Adam (Kingma & Ba, 2015)	0.0623	0.0253	0.0890	0.0452	0.1270	0.0657	0.0773	0.0377
	AdamW (Loshchilov & Hutter, 2019)	0.0619	0.0252	0.0954	0.0481	0.1275	0.0652	0.0768	0.0375
	AdamD (Ding et al., 2025)	0.0623	0.0253	0.0954	0.0481	0.1270	0.0657	0.0771	0.0379
	FAL (Qiu et al., 2025)	0.0645	0.0258	0.1028	0.0489	0.1272	0.0650	0.0780	**0.0382**
	Helen (Zhu et al., 2024)	0.0637	0.0259	0.0954	0.0485	**0.1295**	**0.0661**	0.0779	0.0381
	FUMO (restricted budget)	**0.0671**	**0.0277**	**0.1064**	**0.0494**	0.1283	0.0656	**0.0785**	**0.0382**
GRU4Rec	Adam (Kingma & Ba, 2015)	0.0617	0.0244	0.0780	0.0320	0.0975	0.0458	0.0603	0.0265
	AdamW (Loshchilov & Hutter, 2019)	0.0630	0.0250	0.0807	0.0345	0.1013	**0.0500**	0.0614	0.0271
	AdamD (Ding et al., 2025)	0.0622	0.0250	0.0780	0.0320	0.0999	0.0464	0.0603	0.0265
	FAL (Qiu et al., 2025)	0.0612	0.0243	0.0706	0.0332	0.1006	0.0474	0.0567	0.0252
	Helen (Zhu et al., 2024)	0.0634	0.0254	0.0817	0.0366	0.1024	0.0474	0.0596	0.0260
	FUMO (restricted budget)	**0.0652**	**0.0256**	**0.0890**	**0.0400**	**0.1051**	0.0489	**0.0625**	**0.0285**
FMLPRec	Adam (Kingma & Ba, 2015)	0.0716	0.0286	0.0972	0.0461	0.1376	0.0679	0.0853	0.0416
	AdamW (Loshchilov & Hutter, 2019)	0.0724	0.0296	0.0991	0.0444	0.1376	0.0682	0.0848	0.0413
	AdamD (Ding et al., 2025)	0.0729	0.0298	0.0972	0.0461	0.1381	0.0684	0.0853	0.0411
	FAL (Qiu et al., 2025)	0.0725	0.0294	0.0862	0.0409	**0.1400**	**0.0707**	0.0839	0.0404
	Helen (Zhu et al., 2024)	0.0732	0.0299	0.0972	0.0471	0.1358	0.0690	0.0862	**0.0421**
	FUMO (restricted budget)	**0.0741**	**0.0301**	**0.1138**	**0.0504**	0.1366	0.0688	**0.0864**	0.0420
BSARec	Adam (Kingma & Ba, 2015)	0.0705	0.0286	0.1046	0.0492	0.1341	0.0676	0.0844	0.0412
	AdamW (Loshchilov & Hutter, 2019)	0.0721	0.0293	0.1101	**0.0534**	0.1346	0.0671	0.0832	0.0402
	AdamD (Ding et al., 2025)	0.0722	0.0289	0.1046	0.0495	0.1342	0.0674	0.0835	0.0403
	FAL (Qiu et al., 2025)	0.0692	0.0282	0.1000	0.0493	0.1338	0.0679	0.0841	0.0409
	Helen (Zhu et al., 2024)	0.0718	0.0292	0.1064	0.0519	0.1382	**0.0693**	0.0853	**0.0419**
	FUMO (restricted budget)	**0.0755**	**0.0308**	**0.1174**	0.0513	**0.1388**	0.0686	**0.0856**	0.0407
DuoRec	Adam (Kingma & Ba, 2015)	0.0708	0.0291	0.0844	0.0427	0.1300	0.0663	0.0783	0.0381
	AdamW (Loshchilov & Hutter, 2019)	0.0740	0.0304	0.0908	0.0448	0.1291	0.0659	0.0783	0.0385
	AdamD (Ding et al., 2025)	0.0690	0.0287	0.0844	0.0427	0.1300	0.0663	**0.0788**	0.0386
	FAL (Qiu et al., 2025)	0.0718	0.0297	0.0789	0.0384	0.1293	0.0651	0.0768	0.0371
	Helen (Zhu et al., 2024)	0.0710	0.0294	0.0927	0.0452	**0.1304**	**0.0667**	0.0783	**0.0387**
	FUMO (restricted budget)	**0.0752**	**0.0306**	**0.1000**	**0.0501**	0.1285	0.0655	0.0786	0.0377
SPRINT	Adam (Kingma & Ba, 2015)	0.0627	0.0253	0.0954	0.0457	0.1292	0.0660	0.0762	0.0376
	AdamW (Loshchilov & Hutter, 2019)	0.0624	0.0256	0.0890	0.0451	**0.1306**	**0.0663**	0.0773	0.0374
	AdamD (Ding et al., 2025)	0.0633	0.0260	0.0917	0.0447	0.1298	0.0662	0.0760	0.0374
	FAL (Qiu et al., 2025)	0.0637	0.0264	0.0881	0.0439	0.1294	0.0649	0.0789	**0.0386**
	Helen (Zhu et al., 2024)	0.0633	0.0262	0.1018	0.0477	0.1278	0.0661	0.0781	**0.0386**
	FUMO (restricted budget)	**0.0664**	**0.0272**	**0.1055**	**0.0501**	0.1297	0.0660	**0.0791**	0.0385

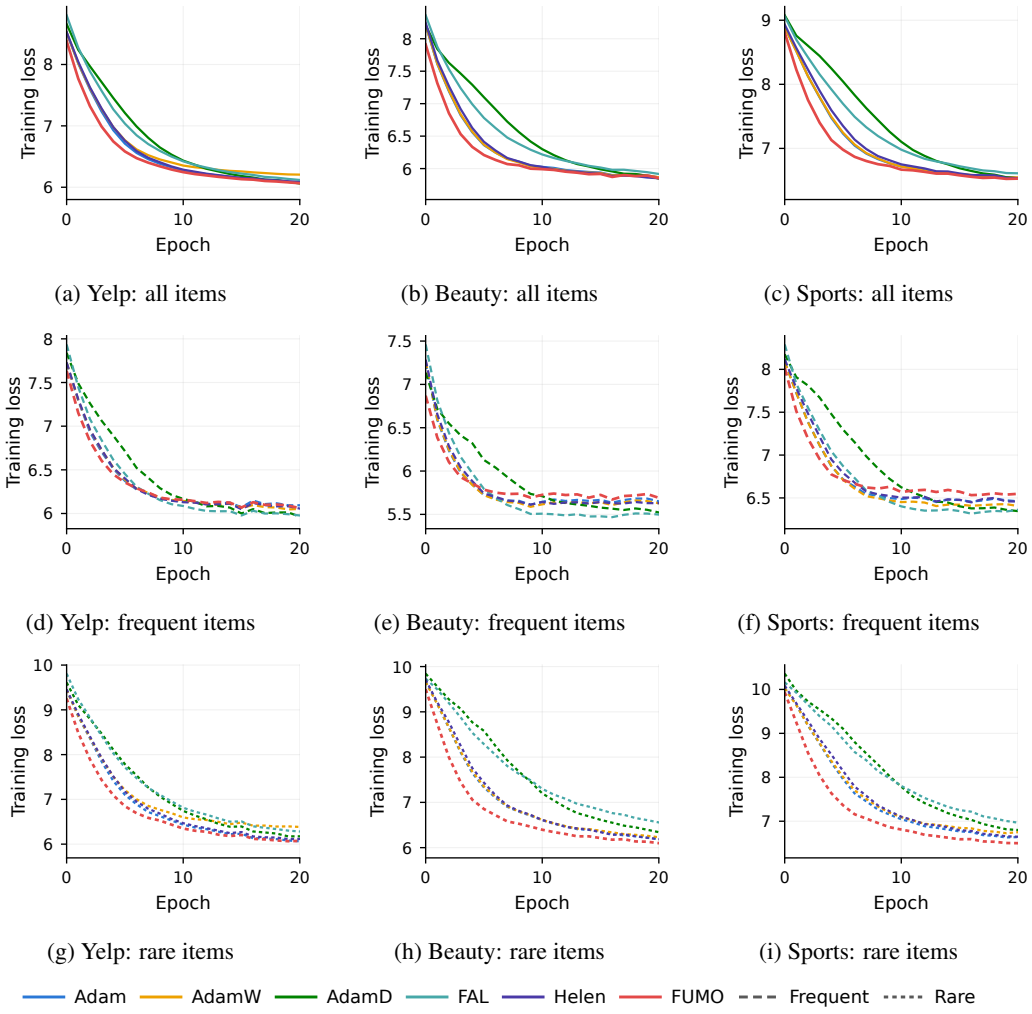


Figure 3: Training cross entropy for SASRec, recomputed at each checkpoint with dropout disabled. Columns are Yelp, Beauty, and Sports; the rows show all (solid), frequent (dashed), and rare items (dotted), respectively. FUMO's early loss advantage comes from the rare items. Vertical scales differ across panels.

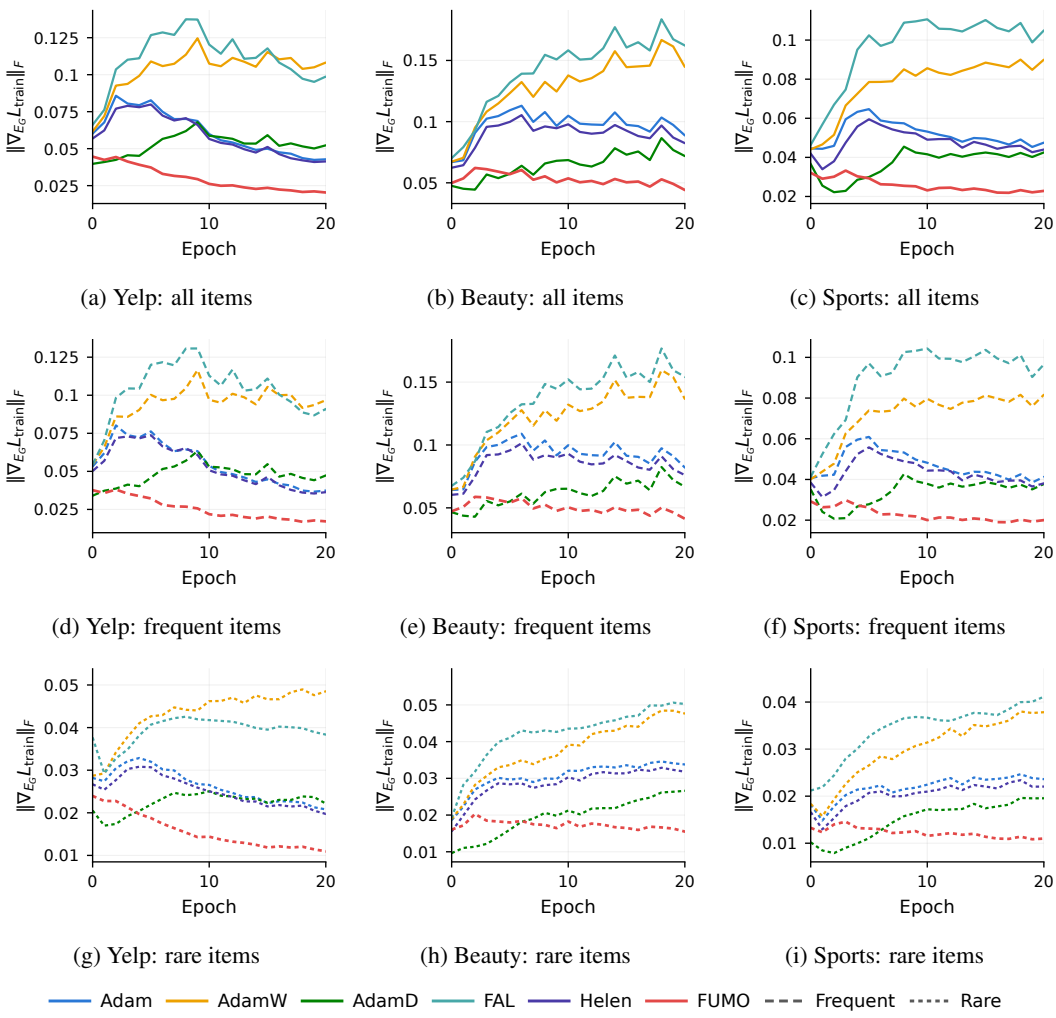


Figure 4: Embedding-gradient norms of the overall training cross entropy for SASRec. Columns are Yelp, Beauty, and Sports; the rows show all (solid), frequent (dashed), and rare items (dotted), respectively. Vertical scales differ across panels. Group norms are unnormalized Frobenius norms, so comparisons hold within an item group.

early rise observed with Adam, Helen, AdamW, and FAL, and its norms generally decrease over training. Adam and Helen decline or plateau after their initial peaks, whereas AdamW and FAL remain substantially higher. AdamD starts below FUMO but subsequently rises above it. By the end of the displayed interval, FUMO has the smallest norm on all three datasets.

B.4 EFFECT OF FUMO ON POPULARITY BIAS

FUMO gives rare items larger relative recommendation updates to address the unequal learning progress in Section 3.2. Improving rare-item accuracy, however, does not necessarily increase their exposure in the final ranked lists. We therefore compare the popularity of recommendations produced by Adam and FUMO across all six models and four datasets.

Metrics. Following Yang et al. (2026), we use long-tail exposure (Fair) and Average Recommendation Popularity (ARP), both evaluated at a cutoff of 20 to match our accuracy metrics. Let $\mathcal{U}$ denote the evaluated users, $\mathcal{I}$ the item catalog excluding padding, and R_u^{20} the set of 20 items recommended to user u after excluding observed interactions, as in Appendix B.2. Let $\alpha = 0.8$ be the tail fraction and $\mathcal{I}_{\text{tail}}$ the $\lfloor \alpha|\mathcal{I}| \rfloor$ items with the smallest counts. Define

$$\text{Fair @20} = \frac{1}{20|\mathcal{U}|} \sum_{u \in \mathcal{U}} \sum_{i \in R_u^{20}} \mathbb{1}_{i \in \mathcal{I}_{\text{tail}}}, \tag{16}$$

$$\text{ARP @20} = \frac{1}{20|\mathcal{U}|} \sum_{u \in \mathcal{U}} \sum_{i \in R_u^{20}} n_i. \tag{17}$$

Fair@20 measures the fraction of recommendation slots occupied by tail items; higher values indicate greater tail exposure. ARP@20 measures the average occurrence count of recommended items; lower values indicate less emphasis on popular items. These metrics describe exposure rather than relevance, and neither alone establishes that recommendations match users' preferences. Since ARP uses raw counts, its scale depends on the dataset's item frequencies.

Overall pattern. Table 5 shows that FUMO increases Fair@20 in 17 of the 24 model–dataset pairs and reduces ARP@20 in 16. Beauty shows the most consistent pattern: tail exposure increases for all six models, and average popularity decreases for five. Some improvements are substantial. On Yelp, DuoRec's Fair@20 rises from 0.0904 to 0.1894, a gain of 9.90 percentage points, while ARP@20 falls from 90.28 to 78.03 (13.6%). On Sports, GRU4Rec's Fair@20 rises from 0.1487 to 0.2335, a gain of 8.48 percentage points, while ARP@20 falls from 139.35 to 114.67 (17.7%). These results show that frequency-aware optimization can substantially increase tail exposure while reducing the average popularity of recommended items.

Mean results by model and dataset. The Mean rows in Table 5 average over the four datasets for each model, while Table 6 averages over the six models for each dataset. Both tables report arithmetic means for each optimizer. Each result receives equal weight; the model-level ARP means summarize raw counts across datasets rather than a normalized measure of bias. Averaged over datasets, FUMO lowers ARP@20 for every model and raises Fair@20 for five of the six models, with SPRINT the exception. DuoRec has the largest mean Fair@20 gain, from 0.2256 to 0.2643. Averaged over models, both metrics improve on Yelp, Beauty, and Sports. LastFM is the exception: mean Fair@20 changes from 0.4574 to 0.4568, while mean ARP@20 rises from 34.20 to 34.80. Thus, the average results favor FUMO across most models and datasets, while retaining a clear dependence on the dataset.

Model and dataset dependence. The effect is not uniform. On LastFM, only SPRINT improves both metrics; SASRec, GRU4Rec, and FMLPRec instead show lower Fair@20 and higher ARP@20. Both metrics also worsen for GRU4Rec on Yelp and SPRINT on Sports. Five other pairs move in opposite directions on the two metrics. For example, DuoRec on Beauty increases Fair@20 from 0.2669 to 0.2752 but also increases ARP@20 from 72.98 to 73.17. Conversely, FMLPRec on Sports reduces ARP@20 from 99.46 to 95.10 while Fair@20 decreases from 0.2512 to 0.2472. Such outcomes are possible because Fair counts membership in a fixed tail group, whereas ARP also reflects popularity differences within both groups. The reported aggregate values do not identify which individual items account for these changes.

Table 5: Popularity of recommendations under Adam and FUMO across six models and four datasets. Fair@20 is the exposure share of the bottom 80% of items by frequency; ARP@20 is the average occurrence count of recommended items. Bold marks the better value within each row.

		Fair@20 ↑		ARP@20 ↓	
Model	Dataset	Adam	FUMO	Adam	FUMO
SASRec	Yelp	0.2233	**0.2377**	73.70	**71.09**
	LastFM	**0.4602**	0.4595	**33.66**	34.74
	Beauty	0.3022	**0.3407**	72.11	**66.21**
	Sports	0.2753	**0.2853**	100.34	**96.41**
	Mean	0.3153	**0.3308**	69.95	**67.11**
GRU4Rec	Yelp	**0.1684**	0.1463	**80.97**	84.77
	LastFM	**0.4637**	0.4264	**35.48**	37.18
	Beauty	0.2731	**0.2944**	78.34	**72.94**
	Sports	0.1487	**0.2335**	139.35	**114.67**
	Mean	0.2635	**0.2752**	83.54	**77.39**
FMLPRec	Yelp	0.2624	**0.2666**	66.00	**63.04**
	LastFM	**0.4773**	0.4738	**33.20**	33.57
	Beauty	0.2915	**0.3152**	69.94	**66.51**
	Sports	**0.2512**	0.2472	99.46	**95.10**
	Mean	0.3206	**0.3257**	67.15	**64.56**
BSARec	Yelp	0.2244	**0.2523**	70.94	**65.90**
	LastFM	0.4894	**0.4983**	**31.97**	31.98
	Beauty	0.3034	**0.3173**	69.85	**67.28**
	Sports	0.2485	**0.2499**	102.46	**100.33**
	Mean	0.3164	**0.3295**	68.81	**66.37**
DuoRec	Yelp	0.0904	**0.1894**	90.28	**78.03**
	LastFM	0.3763	**0.3994**	**37.96**	38.53
	Beauty	0.2669	**0.2752**	**72.98**	73.17
	Sports	0.1689	**0.1930**	115.07	**109.84**
	Mean	0.2256	**0.2643**	79.07	**74.89**
SPRINT	Yelp	**0.2536**	0.2436	71.34	**70.88**
	LastFM	0.4774	**0.4836**	32.91	**32.80**
	Beauty	0.3421	**0.3538**	65.16	**62.43**
	Sports	**0.2807**	0.2638	**97.50**	97.76
	Mean	**0.3385**	0.3362	66.73	**65.97**

Table 6: Mean popularity metrics grouped by dataset, averaging over six models with equal weight for each constituent result. Bold marks the better mean within each row.

	Mean Fair@20 ↑		Mean ARP@20 ↓	
Dataset	Adam	FUMO	Adam	FUMO
Yelp	0.2038	**0.2227**	75.54	**72.29**
LastFM	**0.4574**	0.4568	**34.20**	34.80
Beauty	0.2965	**0.3161**	71.40	**68.09**
Sports	0.2289	**0.2455**	109.03	**102.35**

Implications. FUMO does not guarantee a reduction in popularity bias: its effect on exposure depends on both the model and the dataset, and its update rule does not directly constrain the final recommendation lists. Nevertheless, the results establish FUMO as a strong optimizer for sequential recommendation. It improves both HR@20 and NDCG@20 over Adam in all 24 model–dataset pairs (Section 5.2), accelerates rare-item learning in the examined training trajectories (Section 5.3), and improves both mean popularity metrics on three of the four datasets. These benefits also extend to models with auxiliary objectives, without changing their architecture or loss. Its practical strength is therefore the combination of consistent accuracy gains and improvements in alleviating popularity bias.

B.5 Ablation Study

We examine whether FUMO's accuracy gains can be recovered by a larger embedding learning rate alone, and how they depend on frequency scaling, weight decay, and the treatment of auxiliary gradients. We conduct the ablation study on SASRec and DuoRec.

Variants. Following the notation of Section 4, the full method uses $\rho_i = \gamma\phi_r(n_i)$ and $\lambda_i = \lambda\phi_r(n_i)$, where ϕ_r is the frequency profile in Equation (4). We consider six variants:

- **A1. Adam with a larger embedding learning rate.** Use standard Adam with embedding learning rate $\gamma\eta$ and learning rate η for the remaining parameters, without weight decay. This tests a uniform increase in the embedding step size.
- **A2. No frequency-aware gradient scaling.** Set $\rho_i = 1$ for every item, retaining frequency-aware weight decay.
- **A3. No frequency-aware weight decay.** Set the decay frequency multiplier to one for every item, retaining frequency-aware gradient scaling. In the notation of Equation (7), this means $\lambda_i = \lambda$; the base decay coefficient is unchanged.
- **A4. Inverse-square-root frequency profile.** Replace $\phi_r(n)$ by $\phi(n) = 1/\sqrt{n}$ for $n \geq 1$, following the inverse-square-root frequency dependence of FA-SGD (Li et al., 2022). This substitutes the frequency profile within FUMO.
- **A5. AdamW-style weight decay.** Replace the AdamD-style decay (Ding et al., 2025) in Equation (7) by the shrinkage $\eta\lambda_i\mathbf{1}$, as in AdamW (Loshchilov & Hutter, 2019). Thus, decay is no longer divided by $\sqrt{\widehat{\boldsymbol{v}}_i^{(t)}} + \epsilon$, where $\widehat{\boldsymbol{v}}_i^{(t)}$ is the bias-corrected second moment and ϵ the numerical stabilizer. Frequency dependence and the loss update are retained.
- **A6. Scale the summed gradient (DuoRec only).** Let $\boldsymbol{g}_{\mathrm{rec},i}^{(t)}$ and $\boldsymbol{g}_{\mathrm{aux},i}^{(t)}$ denote the recommendation and auxiliary gradients. Replace the first-moment input $\rho_i\boldsymbol{g}_{\mathrm{rec},i}^{(t)} + \boldsymbol{g}_{\mathrm{aux},i}^{(t)}$ by $\rho_i(\boldsymbol{g}_{\mathrm{rec},i}^{(t)} + \boldsymbol{g}_{\mathrm{aux},i}^{(t)})$. The second moment still uses the unscaled total gradient. This variant tests whether preserving the auxiliary gradient's base coefficient matters when $\mathcal{L}_{\mathrm{aux}} \neq 0$; for SASRec the two inputs coincide.

Uniform acceleration and frequency scaling. Table 7 reports results for both models on all four datasets. Increasing Adam's embedding learning rate alone (A1) falls below FUMO on both metrics in all eight model–dataset pairs. For example, on Yelp its HR@20 is 0.0617 for SASRec and 0.0649 for DuoRec, compared with 0.0689 and 0.0760 for FUMO. Relative to Adam, A1 has mixed effects: it improves both metrics on LastFM for both models but lowers both on Yelp. A uniform increase in the embedding learning rate therefore does not recover FUMO's gains in these experiments. Removing recommendation scaling (A2) lowers HR@20 in all eight pairs and NDCG@20 in seven; the exception is DuoRec on Beauty (0.0673 versus 0.0671). The inverse-square-root profile (A4) falls below FUMO on both metrics in all eight pairs. The largest HR@20 reductions occur for DuoRec on Yelp (0.0760 to 0.0565) and Sports (0.0792 to 0.0567). These comparisons support the proposed frequency scaling at the reported settings, although its benefit varies across models and datasets.

The role of weight decay. Uniform decay (A3) falls below FUMO in 15 of the 16 metric comparisons; the exception is DuoRec's HR@20 on Yelp, where A3 reaches 0.0762 compared with 0.0760 for FUMO. AdamW-style decay (A5) falls below FUMO on both metrics in all eight model–dataset

Table 7: Ablation test accuracy. A1 uses no weight decay. A1–A6 are defined in Appendix B.5. Bold marks the highest displayed value in each model–dataset–metric column across all rows. A6 applies only to DuoRec.

	Yelp		LastFM		Beauty		Sports	
Variant	HR@20	NDCG@20	HR@20	NDCG@20	HR@20	NDCG@20	HR@20	NDCG@20
SASRec								
Adam	0.0623	0.0253	0.0890	0.0452	0.1270	0.0657	0.0773	0.0377
FUMO	**0.0689**	**0.0282**	**0.1037**	**0.0504**	**0.1293**	**0.0662**	**0.0785**	**0.0382**
A1: Adam, embedding LR $\times\gamma$	0.0617	0.0252	0.0936	0.0454	0.1260	0.0660	0.0771	0.0376
A2: No gradient scaling	0.0645	0.0262	0.0945	0.0473	0.1282	0.0655	0.0776	0.0378
A3: Uniform decay	0.0656	0.0266	0.0899	0.0439	0.1253	0.0646	0.0752	0.0368
A4: Inverse square root	0.0626	0.0247	0.0872	0.0428	0.1225	0.0628	0.0764	0.0369
A5: AdamW-style decay	0.0616	0.0249	0.0899	0.0439	0.1257	0.0647	0.0754	0.0365
DuoRec								
Adam	0.0708	0.0291	0.0844	0.0427	0.1300	0.0663	0.0783	0.0381
FUMO	0.0760	**0.0316**	**0.1000**	**0.0501**	**0.1313**	0.0671	**0.0792**	**0.0388**
A1: Adam, embedding LR $\times\gamma$	0.0649	0.0268	0.0927	0.0434	0.1288	0.0663	0.0774	0.0386
A2: No gradient scaling	0.0735	0.0304	0.0844	0.0415	0.1307	**0.0673**	0.0769	0.0375
A3: Uniform decay	**0.0762**	0.0315	0.0982	0.0474	0.1283	0.0653	0.0778	0.0379
A4: Inverse square root	0.0565	0.0224	0.0881	0.0440	0.1215	0.0617	0.0567	0.0265
A5: AdamW-style decay	0.0712	0.0293	0.0991	0.0477	0.1309	0.0655	0.0778	0.0380
A6: Scale summed gradient	0.0760	0.0314	0.0945	0.0472	0.1309	0.0667	0.0780	0.0382

pairs. The two variants have no consistent ordering: for example, uniform decay performs better on DuoRec–Yelp, whereas AdamW-style decay performs better on SASRec–Beauty. These results generally favor frequency-dependent decay and consistently favor adaptive scaling at the reported settings. The AdamW-style variant also changes the effective shrinkage by removing the adaptive denominator, so this comparison does not establish that the same ordering would hold after retuning the decay strength.

Keeping auxiliary gradients at their base scale. Scaling the summed gradient (A6) lowers DuoRec's NDCG@20 relative to FUMO on all four datasets and HR@20 on three. On Yelp, HR@20 ties at the displayed precision (0.0760), while NDCG@20 decreases from 0.0316 to 0.0314. On LastFM, HR@20 decreases from 0.1000 to 0.0945 and NDCG@20 from 0.0501 to 0.0472, a relative reduction of 5.8%. This pattern supports retaining the auxiliary gradient's base coefficient in DuoRec at the reported settings, although the effect on Yelp is small.

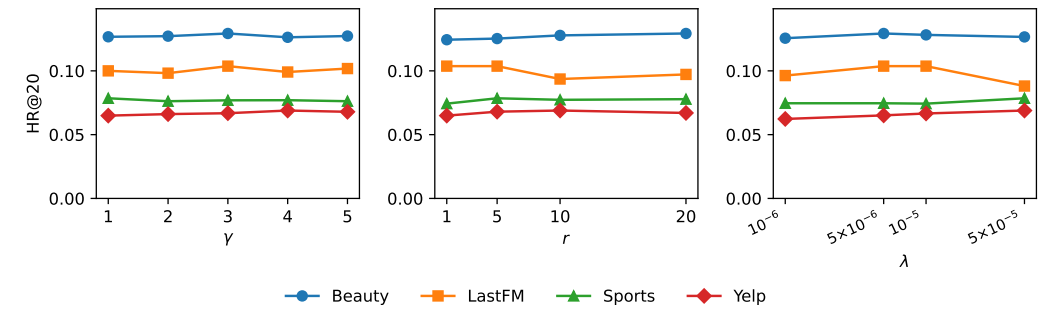
B.6 HYPERPARAMETER SENSITIVITY

We examine how SASRec's and GRU4Rec's recommendation accuracy changes with three FUMO hyperparameters: the global step size multiplier γ, the frequency-scale parameter r, and the base weight-decay coefficient λ, introduced in Section 4. Figure 5 reports HR@20 and NDCG@20 on Yelp, LastFM, Beauty, and Sports. We vary one hyperparameter at a time while holding the other two fixed. GRU4Rec generally favors larger step multipliers than SASRec. Both models favor a larger frequency scale on Beauty, a smaller scale on LastFM, and stronger decay on Yelp and Sports. Nonmonotonic fluctuations are particularly pronounced for GRU4Rec on LastFM and less evident on the other datasets or with SASRec, but this pattern alone does not establish that FUMO is generally highly sensitive to its hyperparameters.

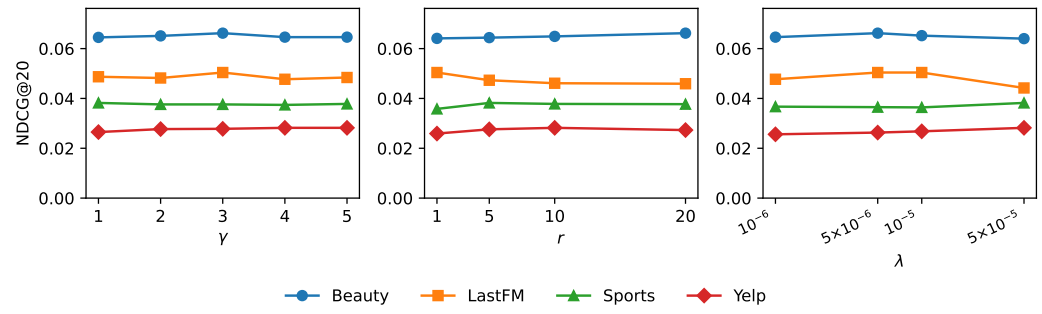
Global step size multiplier. The preferred multiplier depends on both the model and dataset. SASRec attains its highest values on both metrics at $\gamma = 3$ on Beauty and LastFM, $\gamma = 1$ on Sports, and $\gamma = 4$ on Yelp. GRU4Rec instead favors $\gamma = 5$ on Beauty, LastFM, and Yelp. On Beauty, increasing γ from 1 to 5 raises GRU4Rec's HR@20 from 0.0964 to 0.1052 and NDCG@20 from 0.0454 to 0.0493, although neither metric improves monotonically. The response is also nonmonotonic on LastFM, where GRU4Rec's NDCG@20 ranges from 0.0314 to 0.0400, compared with a range of at most 0.0027 in each SASRec sweep. On Sports, GRU4Rec's HR@20 peaks at $\gamma = 5$, while its NDCG@20 peaks at $\gamma = 3$. Larger multipliers can therefore help, but do not provide a consistent improvement across the tested settings.

Frequency scale. Both models improve throughout the tested range on Beauty. At $r = 20$, SASRec reaches HR@20 of 0.1293 and NDCG@20 of 0.0662, while GRU4Rec reaches 0.1052 and 0.0493. Their responses differ on the other datasets. On Sports, SASRec peaks at $r = 5$ on both metrics; GRU4Rec's HR@20 increases from 0.0563 at $r = 1$ to 0.0638 at $r = 20$, while its NDCG@20 remains at 0.0285 for $r \in \{5, 10, 20\}$. On Yelp, SASRec favors $r = 10$ and GRU4Rec favors $r = 5$. LastFM favors smaller scales: SASRec performs best at $r = 1$ and GRU4Rec at $r = 5$. Since r changes frequency scaling in both the recommendation update and weight decay, these curves reflect their combined effect and do not identify a shared optimum across datasets.

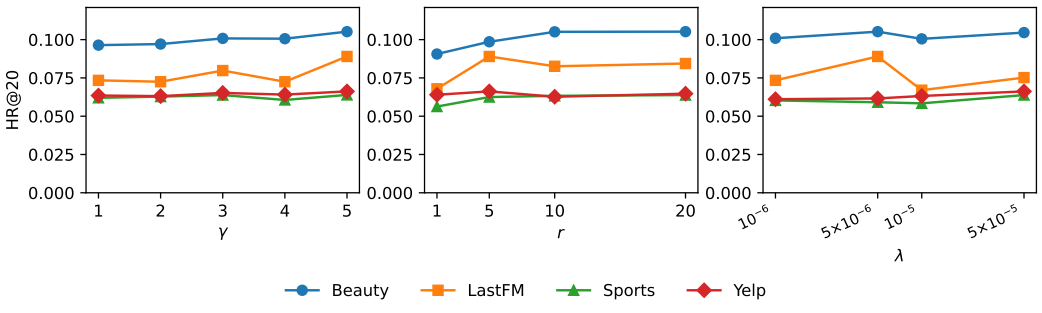
Weight decay. Yelp benefits from stronger decay for both models throughout the tested range. At $\lambda = 5 \times 10^{-5}$, SASRec reaches HR@20 of 0.0689 and NDCG@20 of 0.0282, while GRU4Rec reaches 0.0662 and 0.0263. Both models also attain their highest values on Sports at this largest tested coefficient, although GRU4Rec's HR@20 initially decreases as λ increases. On Beauty, both models favor 5×10^{-6}. LastFM is more sensitive: SASRec's HR@20 and NDCG@20 fall from 0.1037 and 0.0504 at 10^{-5} to 0.0881 and 0.0442 at 5×10^{-5}. GRU4Rec peaks at 5×10^{-6}, with HR@20 of 0.0890 and NDCG@20 of 0.0400, but falls to 0.0670 and 0.0314 at 10^{-5} before partially recovering at 5×10^{-5}. These results support selecting the decay coefficient for the model and dataset rather than assuming that stronger decay is always beneficial.



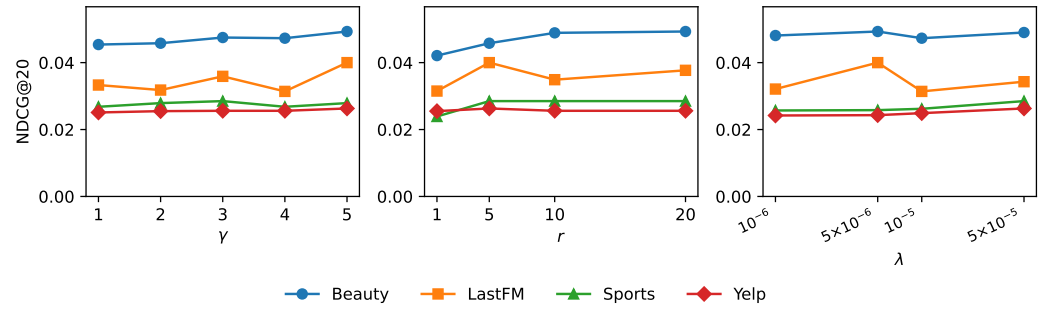
(a) SASRec: HR@20.



(b) SASRec: NDCG@20.



(c) GRU4Rec: HR@20.



(d) GRU4Rec: NDCG@20.

Figure 5: Hyperparameter sensitivity of FUMO with SASRec and GRU4Rec on four datasets. Columns vary the global step size multiplier γ, frequency scale r, and base weight-decay coefficient λ, respectively. The λ axis is logarithmic.