Leveraging Uniformity of Normalized Embeddings for Sequential Recommendation

Hyunsoo Chung
Omnious AI
hyunsoo.chung@omnious.ai

Jungtaek Kim
University of Pittsburgh
jungtaek.kim@pitt.edu

Abstract

Pointwise loss is one of the most widely adopted yet practical choices for training sequential recommendation models. Aside from their successes, only limited studies leverage normalized embeddings in their optimization, which has been actively explored and proven effective in various machine learning fields. However, we observe that the naïve adoption of normalization hinders the quality of a learned recommendation policy. In particular, we argue that the clusterization of embeddings on a unit hypersphere triggers such performance degradation. To alleviate this issue, we propose a novel training objective that enforces the uniformity of embeddings while learning the recommendation policy. We empirically validate our method on sequential recommendation tasks and show superior performance improvements compared to other approaches without normalization.

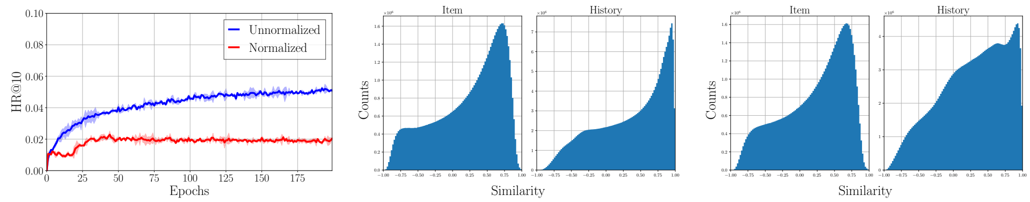
1 Introduction

In sequential recommendation, learning robust feature representations of each user's interaction histories and items lies at the heart of most sequential recommendation systems. A recommendation model generally learns a mapping function that projects users and items into latent vectors defined on the shared embedding space of the same dimension, where a recommendation score is then calculated via the inner product of latent vectors for a pair of history and item. In the realm of the advances of neural networks [28, 5, 15], diverse approaches yield architectural improvements and consequently enhance the quality of the learned recommendation policy [28, 26, 36, 35, 9, 31]. In terms of core training objectives, however, comparably less studies have been suggested such that either pointwise [1, 9] or pairwise loss [25, 24] is usually adopted. While such objectives differ in their forms, both losses utilize unnormalized embeddings in common, thereby intensifying a popularity bias among recommended items [2, 23, 14].

On the other hand, use of normalized representations is the de facto standard due to its improved performance and robustness in a wide variety of applications in computer vision and natural language processing [8, 3, 7, 22]. Despite their successes, we observe that the naïve adoption of such embedding normalization leads to significant performance degradation. In this work, we first empirically analyze the cause of the aforementioned training failure and subsequently introduce a novel method that addresses such a limitation. Specifically, we argue that the clusterization of both item and history embeddings during the training process deteriorates the resulting performance.

To tackle this issue on the clusterization of representations, we introduce a new training objective that prevents the bias of normalized embeddings where the recommendation policy is simultaneously learned. On top of the original pointwise recommendation loss, we introduce a novel regularization term, which is motivated by the uniformity constraint [30], to relax the skewness of learned embeddings. The model consequently maintains maximal information required to recommend the most relevant items depending on each user's history within normalized representations. We validate the

37th Conference on Neural Information Processing Systems (NeurIPS 2023).



(a) Training progress (b) Early training stage (c) Late training stage

Figure 1: Experiments on the clusterization issue with the naïve adoption of normalization during training. Figure 1(a) illustrates performance difference between unnormalized and normalized embeddings. Figures 1(b) and 1(c) represent pairwise cosine similarities of history and item embeddings.

proposed loss on four different sequential recommendation benchmarks and demonstrate superior performance improvements over different approaches with unnormalized embeddings.

2 Preliminaries

Problem Formulation. Let a set of M users be $\mathcal{U} = \{u_1, \ldots, u_M\}$ and a set of N items be $\mathcal{I} = \{i_1, \ldots, i_N\}$. A sequential recommendation task requires capturing a dynamic user behavior from a previous interaction history. By utilizing the previous interaction history of a user u, denoted as $h_u = \{i^u_1, \ldots, i^u_t\}$, where i^u_j is jth item for u and eventually $i^u_1, \ldots, i^u_t$ are chronologically ordered, a goal of sequential recommendation is to select the most relevant next item $i^u_{t+1} \in \mathcal{I}$ to u.

Embedding-based Recommendation. Learning a parametric function to embed original vectors of users and items to their hidden representations has steadily proven its effectiveness in tackling a recommendation problem [9]. Given a history h_u and an item i_k, a model, parameterized by $\boldsymbol{\theta}$, first projects h_u and i_k onto the vectors of the same dimension h'_u and i'_k, respectively. To process h_u, which can be considered as a sequence, a model that can handle sequences, e.g., Gated Recurrent Unit (GRU) [4] and Transformer [28], is employed. Then, a recommendation score between h'_u and i'_k is calculated through an inner product of the two vectors, which is given by the following:

$$\hat{s}_{uk} = {h'_u}^\top i'_k. \tag{1}$$

Training Objective. Pointwise loss is one of the most iconic training objectives to learn adequate embeddings of users and items [1, 34]. Among many variants available, we adopt a conventional strategy of predicting the next item. The corresponding objective is employed to train a recommendation model with a dataset $\mathcal{D}$ consisting of (u, i, j) where an item i is observed (i.e., positive) and an item j is unobserved (i.e., negative) to a user u. Then, the recommendation loss $\mathcal{L}_{\text{rec}}$ is defined as the following form based on a binary cross-entropy:

$$\mathcal{L}_{\text{rec}}(\mathcal{D}; \boldsymbol{\theta}) = -\sum_{(u,i,j)\in\mathcal{D}} \log \sigma(\hat{s}_{ui}; \boldsymbol{\theta}) + \log(1 - \sigma(\hat{s}_{uj}; \boldsymbol{\theta})), \tag{2}$$

where $\hat{s}_{ui}$ and $\hat{s}_{uj}$ are the predicted scores of items i and j, respectively, and σ is a sigmoid function.

3 Clusterization of Embeddings

Motivated by the success of embedding normalization in other machine learning fields [29, 33, 6], we analyze its effects on a general recommendation task. Given the two latent representations, h'_u and i'_k, we normalize each vector such that they reside on the surface of a unit hypersphere:

$$\bar{h}'_u = \frac{h'_u}{\|h'_u\|_2} \quad \text{and} \quad \bar{i}'_k = \frac{i'_k}{\|i'_k\|_2}. \tag{3}$$

We thus remove magnitude information from each embedding and alternatively measure a score between two vectors using the cosine similarity between two normalized vectors. To validate its effectiveness, we report the resulting recommendation performance by comparing the use of unnormalized embeddings to the use of normalized embeddings where a neural architecture and loss are fixed. The corresponding results are visualized in Figure 1. We find that the naïve adoption of normalization to the training of the recommender system instead leads to a significant performance drop of the recommendation quality; see Figure 1(a). We presume that the normalized representations do not preserve substantial information for recommendation as magnitudes become identical. Inspired by [17], we calculate a pairwise cosine similarity of normalized item and history embeddings independently to validate our hypothesis. In Figure 1(b), we observe that the history embeddings are typically clusterized at the early stage of the training. While the level of skewness decreases as training proceeds, we still find the bias of embeddings, not covering a broad surface of the unit hypersphere. Nevertheless, we see that performance slowly increases as embeddings become less biased as illustrated in Figure 1(c), which is consistent with our hypothesis.

4 Proposed Method

To alleviate the issue aforementioned, we present a novel training objective for recommendation and thoroughly describe rationales behind each component.

First and foremost, our embeddings should be distributed evenly as much as possible on the hypersphere thereby preserving sufficient information for recommendation. Following the uniformity metric proposed in [30], we design a regularization term based on the Gaussian potential kernel over embeddings but with a slight modification. Given a batch of triplets (u, i, j), we first split the batch into two disjoint sets: $\mathcal{D}_H$ consisting of only history embeddings whereas $\mathcal{D}_I$ consisting of only item embeddings. We then calculate a sum of pairwise Gaussian potentials for each set homogeneously:

$$\mathcal{L}_{\text{hom}} = \sum_{x,y\in\mathcal{D}_H} e^{-t\|\bar{h}'_x - \bar{h}'_y\|_2^2} + \sum_{x,y\in\mathcal{D}_I} e^{-t\|\bar{i}'_x - \bar{i}'_y\|_2^2}, \tag{4}$$

where $\bar{h}'$ and $\bar{i}'$ are normalized history and item embeddings, respectively. By minimizing the $\mathcal{L}_{\text{hom}}$, we expect history embeddings less skewed, and the same for item embeddings as well. Additionally, we define a heterogeneous term as a sum of pairwise Gaussian potentials between each history and negative item embedding:

$$\mathcal{L}_{\text{het}} = \sum_{x\in\mathcal{D}_H, y\in\mathcal{D}_J} e^{-t\|\bar{h}'_x - \bar{i}'_y\|_2^2}. \tag{5}$$

where $\mathcal{D}_J$ is a subset of $\mathcal{D}_I$, containing only negative items from the batch. In particular, $\mathcal{L}_{\text{het}}$ sets each history embedding to be generally far from the embeddings of negative items. Combining Equations (4) and (5) with the pointwise loss $\mathcal{L}_{\text{rec}}$, a final form of our objective is defined as follows:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \beta_1\mathcal{L}_{\text{hom}} + \beta_2\mathcal{L}_{\text{het}}. \tag{6}$$

For simplicity, we use particular t, β_1, and β_2; see the appendix for their details. We would like to note that our proposed loss is versatile since the replacement of the recommendation loss (e.g., a pairwise loss instead of a pointwise loss) or the embedding module (e.g., different neural networks) can be readily achieved with minimal effort and no extra modification. Finally, the recommendation score is calculated similarly as the inner product but now between normalized embeddings.

5 Experiments

In this section, we conduct comprehensive experiments to empirically validate the effectiveness of our proposed loss over existing training losses.

5.1 Experimental Setup

Datasets and Evaluation Metrics. We use four publicly available sequential recommendation benchmarks: Beauty, Toys, and Sports categories from the Amazon datasets [20], and the Yelp dataset.[1] Details of preprocessing procedure and resulting statistics for each dataset are illustrated in the appendix. To evaluate the quality of a trained recommendation model, we adopt two common top-K metrics: top-K Hit Ratio (HR@K) and top-K Normalized Discounted Cumulative Gain (NDCG@K). With the trained model, we recommend K items with the highest recommendation scores from the entire item pool. Note that K is set as 10.

[1] `https://www.yelp.com/dataset`

Table 1: Overall performance of different methods. Ten items from the entire item pool are recommended. For each model and benchmark, results in boldface are best performing methods.

Model	Method	Beauty		Toys		Sports		Yelp	
		HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG
GRU4Rec	BCE	0.0369	0.0185	0.0298	0.0164	0.0148	0.0076	0.0273	0.0136
	BPR	0.0472	0.0248	0.0492	0.0276	0.0283	0.0153	0.0368	0.0190
	InfoNCE	0.0365	0.0169	0.0282	0.0141	0.0314	0.0153	0.0482	0.0241
	Our Loss	**0.0702**	**0.0373**	**0.0705**	**0.0385**	**0.0365**	**0.0193**	**0.0665**	**0.0378**
Caser	BCE	0.0348	0.0172	0.0250	0.0125	0.0218	0.0110	0.0251	0.0121
	BPR	0.0332	0.0159	0.0298	0.0140	0.0165	0.0083	0.0644	0.0342
	InfoNCE	0.0421	0.0172	0.0371	0.0149	0.0245	0.0107	0.0643	0.0333
	Our Loss	**0.0544**	**0.0258**	**0.0416**	**0.0174**	**0.0261**	**0.0116**	**0.0658**	**0.0357**
SASRec	BCE	0.0522	0.0278	0.0604	0.0295	0.0301	0.0145	0.0507	0.0278
	BPR	0.0594	0.0261	0.0662	0.0309	0.0337	0.0150	0.0552	0.0326
	InfoNCE	0.0588	0.0261	0.0677	0.0305	0.0367	0.0165	0.0593	0.0334
	Our Loss	**0.0821**	**0.0371**	**0.0896**	**0.0411**	**0.0471**	**0.0214**	**0.0668**	**0.0405**

Baselines. We verify the effectiveness of our proposed loss with three different architectures for embedding backbones; GRU4Rec [11], Caser [27] and SASRec [13]. Within each architecture, we only switch the training objective and measure the quality of the resulting recommendation policy for valid comparisons of losses. Specifically, we adopt BPR loss [25], BCE loss, and InfoNCE loss [21] as baseline methods, all trained without normalization.

5.2 Results and Analyses

Table 1 summarizes the overview of the performance of baselines and our proposed method, with carefully tuned hyperparameters for all configurations in all datasets. We observe that our proposed objective consistently outperforms all baseline methods regardless of the embedding architectures. Such results imply that a model with normalized embeddings can exhibit improved quality in recommendation when properly trained. Specifically, we presume that our method successfully resolves the training failure of the naïve normalization approach.

It is noteworthy that the most effective baseline loss changes with respect to the utilized model architecture. For instance, while the BPR loss shows the most impressive performance with SASRec and GRU4Rec, the InfoNCE loss turns out to be the best for Caser than other baselines. Our loss, on the other hand, surpasses the best performing baseline in all datasets generally by big margin irrespective of the backbone used. Such model-agnostic tendency depicts the robustness and effectiveness of normalized embeddings, especially when combined with our regularization term.

For model comparison, we observe SASRec to be a generally better pipeline than the rest in terms of reported metrics in all datasets. In the Beauty dataset, for example, SASRec achieved almost 17% performance improvement over GRU4Rec and 51% improvement over Caser when trained with our proposed loss. Meanwhile, we discover that performance of our loss in the Yelp dataset tends to be similar regardless of the architecture used. Hence, we argue that the effective choice of architecture varies depending on the data discrepancy of the tested benchmark. Further ablation studies to validate the effectiveness of each component of our method are presented in the appendix.

6 Conclusion

In this work, we focused on applying embedding normalization to the training process of recommender systems. We analyzed the possible cause of performance drop with simple adoption of embedding normalization. To tackle the issue, we proposed a novel uniformity-inspired objective that enhances the quality of recommendation with normalization. Through a set of experiments on public recommendation benchmarks, we empirically validated its effectiveness and robustness compared to existing methods with unnormalized embeddings.

Acknowledgments and Disclosure of Funding

HC acknowledges and thanks the OMNIOUS.AI team for their support and helpful discussions. JK acknowledges support from the MDS-Rely Center, which is funded by the National Science Foundation's Industry–University Cooperative Research Center (IUCRC) program through awards EEC-2052662 and EEC-2052776.

References

[1] Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. Sequential recommendation with graph neural networks. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 378–387, Virtual, 2021. ACM.

[2] Jiawei Chen, Junkang Wu, Jiancan Wu, Xuezhi Cao, Sheng Zhou, and Xiangnan He. Adap-τ: Adaptively modulating embedding magnitude for recommendation. In *Proceedings of the Web Conference (WWW)*, pages 1085–1096, Austin, Texas, USA, 2023. ACM.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1597–1607, Virtual, 2020. JMLR.

[4] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. ACL, 2014.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[6] Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. Representation degeneration problem in training natural language generation models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, New Orleans, Louisiana, USA, 2019. OpenReview.

[7] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 21271–21284, Virtual, 2020. Curran Associates.

[8] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9729–9738, Virtual, 2020. IEEE.

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the Web Conference (WWW)*, pages 173–182, Perth, Australia, 2017. ACM.

[10] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. LightGCN: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 639–648, Virtual, 2020. ACM.

[11] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015.

[12] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 263–272, Pisa, Italy, 2008. IEEE.

[13] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 197–206, Singapore, Singapore, 2018. IEEE.

[14] Dain Kim, Jinhyeok Park, and Dongwoo Kim. Test time embedding normalization for popularity bias mitigation. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, Birmingham, UK, 2023. ACM.

[15] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Palais des Congrès Neptune, Toulon, France, 2018. OpenReview.

[16] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[17] Victor Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Y Zou. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 17612–17625, Virtual, 2022. Curran Associates.

[18] Zhuang Liu, Yunpu Ma, Yuanxin Ouyang, and Zhang Xiong. Contrastive learning for recommender system. *arXiv preprint arXiv:2101.01317*, 2021.

[19] Kelong Mao, Jieming Zhu, Jinpeng Wang, Quanyu Dai, Zhenhua Dong, Xi Xiao, and Xiuqiang He. Simplex: A simple and strong baseline for collaborative filtering. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1243–1252, Gold Coast, Australia, 2021. ACM.

[20] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 43–52, Shanghai, China, 2015. ACM.

[21] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[22] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 8748–8763. PMLR, 2021.

[23] Weijieying Ren, Lei Wang, Kunpeng Liu, Ruocheng Guo, Lim Ee Peng, and Yanjie Fu. Mitigating popularity bias in recommendation with unbalanced interactions: A gradient perspective. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 438–447, Orlando, Florida, USA, 2022. IEEE.

[24] Steffen Rendle and Christoph Freudenthaler. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 273–282, Shenzhen, China, 2014. IEEE.

[25] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 452–461, Montreal, Quebec, Canada, 2009. AUAI Press.

[26] 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 ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1441–1450, Beijing, China, 2019. ACM.

[27] Jiaxi Tang and Ke Wang. Personalized top-N sequential recommendation via convolutional sequence embedding. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 565–573, Singapore, Singapore, 2018. IEEE.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pages 5998–6008, Long Beach, California, USA, 2017. Curran Associates.

[29] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. In *Proceedings of the ACM International Conference on Multimedia (MM)*, pages 1041–1049, New York, New York, USA, 2017. ACM.

[30] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 9929–9939, Virtual, 2020. JMLR.

[31] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 165–174, Paris, France, 2019. ACM.

[32] Jiancan Wu, Xiang Wang, Xingyu Gao, Jiawei Chen, Hongcheng Fu, Tianyu Qiu, and Xiangnan He. On the effectiveness of sampled softmax loss for item recommendation. *arXiv preprint arXiv:2201.02327*, 2022.

[33] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3733–3742, Salt Lake City, Utah, USA, 2018. IEEE.

[34] Huachi Zhou, Qiaoyu Tan, Xiao Huang, Kaixiong Zhou, and Xiaoling Wang. Temporal augmented graph neural networks for session-based recommendations. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 1798–1802, Virtual, 2021. ACM.

[35] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1893–1902, Virtual, 2020. ACM.

[36] 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 Web Conference (WWW)*, pages 2388–2399, Lyon, France, 2022. ACM.

A Dataset Preprocessing

Following the data preparation procedure in [20], we first regard each dataset consisting of implicit feedback only while removing users and items that appear less than five times in total interactions. Then for dataset partitioning, we adopt the conventional *leave-one-out* strategy [13, 26]: the last two interacted items of each user are utilized for validation and test, while the remaining items are used to train the model.

B Hyperparameters

For all tested datasets, we use a batch size as 256, a maximum sequence length as 50, and a dropout rate as 0.5 regardless of the backbones and losses. For the other hyperparamters, we apply grid search to select the best training configurations. Corresponding search spaces are $\{0.0001, 0.0002, 0.0003\}$ for the learning rate, $\{32, 64, 128\}$ for the embedding dimension size, and $\{1, 2, 3\}$ for the number of layers and heads, respectively.

For simplicity, we set coefficients β_1 and β_2 of our proposed loss to a same value β that is chosen from $\{0.05, 0.01, 0.005, 0.001\}$. In addition, we fixed the value of t of the Gaussian potential kernel to 2 throughout all experiments. Finally, we utilize early stopping, so that a model is trained until validation performance does not improve for more than 20 epochs.

C Ablation Studies

We design and conduct extensive ablation studies to thoroughly inspect each component of our loss and the impact of tuning the corresponding hyperparameter.

Table 2: Ablation study of our proposed loss function. Blank space indicates the absence of corresponding term. Metrics are computed only with the SASRec architecture.

Reference	**Normalization**	$\mathcal{L}_{\text{rec}}$	$\mathcal{L}_{\text{hom}}$	$\mathcal{L}_{\text{het}}$	**Beauty**		**Toys**	
					HR	NDCG	HR	NDCG
(a)		✓			0.0522	0.0278	0.0604	0.0295
(b)	✓	✓			0.0233	0.0113	0.0181	0.0093
(c)	✓	✓	✓		0.0786	0.0362	0.0865	0.0394
(d)	✓	✓		✓	0.0416	0.0205	0.0399	0.0204
(e)	✓	✓	✓	✓	0.0821	0.0371	0.0896	0.0411
(f)	✓		✓	✓	0.0028	0.0015	0.0023	0.0011
(g)		✓	✓	✓	0.0534	0.0257	0.0614	0.0301

C.1 Loss Component Analysis

In Table 2, the detailed comparison of the resulting performance trained with different combinations of our loss components is presented. Here, we fix the model architecture to SASRec and regularization coefficient β to the value of 0.05. We observe performance of the original next item prediction loss, denoted by (a), deteriorates significantly when simply adopting the normalization as (b). On the other hand, we see a dramatic performance gain when trained with our proposed loss (e), hence verifying the effectiveness of uniformity-inspired regularization.

While all components contribute to the increased performance, we notice $\mathcal{L}_{\text{hom}}$ most critical for such gain by comparing the performance of (c) to (b). Such result indicate the importance of limiting clusterization of embeddings while training when normalized. Nevertheless, we observe $\mathcal{L}_{\text{het}}$ further improves the quality of recommendations as seen in performance difference between (e) and (d). Finally, we design and report performance of (f) and (g) to verify the importance of normalization and original recommendation loss. We then observe significant performance drop compared to the model trained with complete form of our loss ((e)). Such results indicate that each component of our loss is necessary to achieve the enhancement.

Table 3: Ablation study on the regularization coefficient β. All metrics are compute with only the SASRec architecture. Other hyperparameters are fixed to a same configuration.

β	**Beauty**		**Toys**	
	HR	NDCG	HR	NDCG
0.1	0.0785	0.0358	0.0854	0.0389
0.05	0.0821	0.0371	0.0896	0.0411
0.01	0.0658	0.0314	0.0708	0.0331
0.005	0.0557	0.0268	0.0582	0.0280
0.001	0.0353	0.0168	0.0320	0.0158

C.2 Effects of Regularization Coefficients

To investigate the effect of the regularization parameters β_1 and β_2, we examine the resulting performance by differentiating the value. For simplicity as again, we fix both coefficients to a same value of β and adjust accordingly. Table 3 summarizes the result of our proposed loss on Beauty and Toys dataset with different values of coefficient. We discover the value of 0.05 achieves the highest metric while either value above it or below it degrades the performance. Thus, selection of the coefficient to appropriate value is necessary to enjoy the stable and robust performance with embedding normalization. Otherwise, resulting performance can even be worse than original recommendation loss without the normalization.

D Related Work

In this section, we review several representative approaches for sequential recommendation tasks and attempts at utilizing normalization in recommendation tasks.

D.1 Sequential Recommendation

Traditional work on sequential recommendation builds upon the idea of decomposing users and items into latent vector representations [16, 12]. By utilizing deep neural networks [28] as an embedding module, such methods have achieved enormous performance improvements. GRU4Rec [11] and Caser [27] adopt recurrent neural network and convolution-based embedding modules, respectively. SASRec [13] and BERT4Rec [26] are two representative frameworks that employ Transformer-based architectures. Recently, MLP-based models such as [36] and GNN-based model [10] have been introduced as well for further improvements.

D.2 Normalization in Recommendation

Despite its rarity, there have been continuous but few approaches combining embedding normalization in recommendation tasks. [32] incorporates embedding normalization with the InfoNCE loss and further examines the behavior of the trained recommender. [18] utilizes normalized embeddings in a contrastive loss to overcome the problem of false negatives during sampling. [19] introduces a cosine contrastive loss that operates on normalized embeddings to prevent the intervention of magnitude information. [23] suggests gradient-based embedding adjustment approach that adopts normalization to resolve the popularity bias problem. [2] adaptively adjusts the magnitude of embeddings to improve recommendation performance. [14] proposes a test-time normalization approach to mitigating the popularity bias issue of conventional recommender systems.