An Adversarial Approach to Improve Long-Tail Performance in Neural Collaborative Filtering

Adit Krishnan[†], Ashish Sharma[*], Aravind Sankar[†], Hari Sundaram[†]
[†]University of Illinois at Urbana-Champaign, IL, USA
[*]Microsoft Research, Bangalore, India
[†]{aditk2, asankar3, hs1}@illinois.edu [*]t-asshar@microsoft.com

ABSTRACT

In recent times, deep neural networks have found success in Collaborative Filtering (CF) based recommendation tasks. By parametrizing latent factor interactions of users and items with neural architectures, they achieve significant gains in scalability and performance over matrix factorization. However, the long-tail phenomenon in recommender performance persists on the massive inventories of online media or retail platforms. Given the diversity of neural architectures and applications, there is a need to develop a generalizable and principled strategy to enhance long-tail item coverage.

In this paper, we propose a novel adversarial training strategy to enhance long-tail recommendations for users with Neural CF (NCF) models. The adversary network learns the implicit association structure of entities in the feedback data while the NCF model is simultaneously trained to reproduce these associations and avoid the adversarial penalty, resulting in enhanced long-tail performance. Experimental results show that even without auxiliary data, adversarial training can boost long-tail recall of state-of-the-art NCF models by up to 25%, without trading-off overall performance. We evaluate our approach on two diverse platforms, content tag recommendation in Q&A forums and movie recommendation.

CCS CONCEPTS

• **Information systems → Collaborative filtering**; **Recommender systems**; • **Computing methodologies → Neural networks**;

KEYWORDS

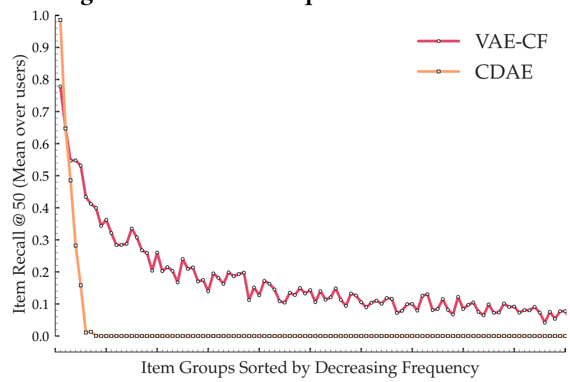
Recommender Systems; Neural Collaborative Filtering; Adversarial Learning; Long-Tail Phenomenon

ACM Reference Format:
Adit Krishnan, Ashish Sharma, Aravind Sankar, Hari Sundaram. 2018. An Adversarial Approach to Improve Long-Tail Performance in Neural Collaborative Filtering. In *Proceedings of The 27th ACM International Conference on Information and Knowledge Management (CIKM '18).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3269206.3269264

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3269264

Figure 1: CDAE[15] and VAE-CF[9] recall for item-groups (decreasing frequency) in MovieLens (*ml-20m*). CDAE overfits to popular item-groups, falls very rapidly. VAE-CF has better long-tail recall due to representational stochasticity.



1 INTRODUCTION

Recommender systems play a pivotal role in sustaining massive product inventories on online media and retail platforms, and reduce information overload on users. Collaborative filtering methods personalize item recommendations based on historic interaction data (implicit feedback setting), with matrix-factorization being the most popular approach [5]. In recent times, NCF methods [3, 9, 15] have transformed simplistic inner-product representations with non-linear interactions, parametrized by deep neural networks. Although performance gains over conventional approaches are significant, a closer analysis indicates skew towards popular items (Figure 3) with ample evidence in the feedback (overfit to popular items), resulting in poor niche (long-tail) item recommendations to users (see fig. 1). This stifles user experience and reduces platform revenue from niche products with high profit margins.

Conventional effort to challenge the long-tail in recommendation has been two-fold [16]. First, integration with neighbor-based models [10] to capture inter-item, inter-user and cross associations in the latent representations and second, incorporating auxiliary data (e.g. item descriptions) to overcome limited feedback [13] or hybrid methods [6, 11]. While neural models readily adapt auxiliary data [8], the association/neighbor-based path is relatively unexplored due to the heterogeneity of representations and architectures.

Given the diversity of NCF architectures and applications [3, 8, 9], architectural solutions may not generalize well. Instead we propose to augment NCF training to levy penalties when the recommender fails to identify suitable niche items for users, given their history and global item co-occurrence. To achieve this, conventional neighbor models employ static pre-computed links between entities [10] to regularize the learned representations. While it is possible to add a similar term to the NCF objective, we aim to learn the association structure rather than imposing it on the model. Towards this goal, we introduce an adversary network to infer the inter-item association structures unlike link-based models, guided by item co-occurrences in the feedback data. The adversary network is trained in tandem with the recommender. It can readily integrate auxiliary data and be extended to model inter-user or cross associations.

For each user, a penalty is imposed on the recommender if the suggested niche items do not correlate with the user's history. The adversary is trained to distinguish the recommender's niche item suggestions against actual item pairings sampled from the data. The more confident this distinction, the higher the penalty imposed. As training proceeds, the adversary learns the inter-item association structure guided by the item pairs sampled from user records while the recommender incorporates these associations, until mutual convergence. In summary, we make the following contributions:

- Unlike conventional neighbor models, our adversary model learns the association structure of entities rather than imposing pre-defined links on the recommender model.
- Our approach is architecture and application agnostic.
- Experimental results on two diverse platforms show substantial gains (by upto 25%) in long-tail item recall for state-of-the-art NCF models while not degrading overall results.

We now present our problem formulation, model details (sec. 2, 3) experimental results (sec. 4), and conclude in sec. 5.

2 PROBLEM DEFINITION

We consider the implicit feedback setting with binary interaction matrix $\mathcal{X} \in \mathbb{Z}_2^{M_{\mathcal{U}} \times M_{\mathcal{I}}}, \mathbb{Z}_2 = \{0, 1\}$ given users $\mathcal{U} = \{u_1, \ldots, u_{M_{\mathcal{U}}}\}$, items $\mathcal{I} = \{i_1, \ldots, i_{M_{\mathcal{I}}}\}$. Items $\mathcal{I}$ are partitioned apriori into two disjoint sets, $\mathcal{I} = \mathcal{I}^{\mathcal{P}}$ (*popular items*) $\cup\, \mathcal{I}^{\mathcal{N}}$ (*niche/long-tail items*) based on their frequency in $\mathcal{X}$. We use the notation $\mathcal{X}_u$ to denote the set of items interacted by $u \in \mathcal{U}$, further split into popular and niche subsets $\mathcal{X}_u^{\mathcal{P}}, \mathcal{X}_u^{\mathcal{N}}$ respectively.

The base neural recommender model $\mathbf{G}$ learns a scoring function $f_{\mathbf{G}}(i \mid u, \mathcal{X}), i \in \mathcal{I}, u \in \mathcal{U}$ to rank items given u's history $\mathcal{X}_u$ and global feedback $\mathcal{X}$, by minimizing CF objective function $O_{\mathbf{G}}$ over recommender $\mathbf{G}$'s parameters θ via stochastic gradient methods. Typically, $O_{\mathbf{G}}$ is composed of a reconstruction loss (analogous to conventional inner product loss [5]) and a suitable regularizer depending on the architecture. We adopt $\mathbf{O_G}$ as a starting point in our training process. Our goal is to enhance the long-tail performance of recommender $\mathbf{G}$ with emphasis on the niche items $\mathcal{I}^{\mathcal{N}}$.

3 MODEL

Most NCF models struggle to recommend niche items with limited click histories, owing to the implicit bias of the reconstruction based objective. Conventional neighbor models [10] apply simplistic pre-defined associations such as Pearson correlation first, and then learn the social representations for recommendation. In contrast, our key insight is that these two tasks are mutually dependent, namely generating item recommendations for user u, and modeling the associations of recommended niche items to his history $\mathcal{X}_u$. The adversarial network paradigm [2] fits our application well, we seek to balance the tradeoff between the popular item biased reconstruction objective against the recall and accuracy of long-tail item recommendations.

Towards the above objective, we introduce the adversary model $\mathbf{D}$ in our learning framework to learn the inter-item association structure in the feedback data and correlate $\mathbf{G}$'s niche item recommendations with popular items in the user's history, $\mathcal{X}_u^{\mathcal{P}}$. We associate $\mathbf{G}$'s niche item recommendations with u's popular item history since niche-popular pairings are the most informative (inter-popular pairs are redundant, inter-niche pairs are noisy). The adversary $\mathbf{D}$ is trained to distinguish "fake" or synthetic pairings of popular and niche items sampled from $\mathcal{X}_u^{\mathcal{P}}$ and $f_{\mathbf{G}}(i \mid u, \mathcal{X})$ respectively, against "real" popular-niche pairs sampled from the global co-occurrence counts in $\mathcal{X}$. The more confident this distinction by $\mathbf{D}$, the stronger the penalty on $\mathbf{G}$. To overcome the applied penalty, $\mathbf{G}$ must produce niche item recommendations that are correlated with the user's history. The model converges when both the synthetic and true niche-popular pairs align with the association structure learned by $\mathbf{D}$. We now formalize the strategy.

True & Synthetic Pair Sampling.

- **True Pairs :** "True" popular-niche pairs $(i^p, i^n) \in \mathcal{I}^{\mathcal{P}} \times \mathcal{I}^{\mathcal{N}}$ are sampled from their global co-occurrence counts in $\mathcal{X}$. To achieve efficiency, we use the alias table method [7] which has $O(1)$ amortized cost when repeatedly drawing samples from the same discrete distribution, compared to $O(\mathcal{I}^{\mathcal{P}} \times \mathcal{I}^{\mathcal{N}})$ for standard sampling. We will denote the true distribution of pairs from $\mathcal{X}$ as $p_{true}(i^p, i^n)$.
- **Synthetic Pairs :** Synthetic pairs $(\tilde{i^p}, \tilde{i^n}) \in \mathcal{I}^{\mathcal{P}} \times \mathcal{I}^{\mathcal{N}}$ are drawn on a per-user basis with $\tilde{i^n} \propto f_{\mathbf{G}}(\tilde{i^n} \mid u, \mathcal{X})$, and $\tilde{i^p}$ randomly drawn from $\mathcal{X}_u^{\mathcal{P}}$. The number of synthetic pairs drawn for each user u is in proportion to $|\mathcal{X}_u^{\mathcal{P}}|$. We denote the resulting synthetic pair distribution $p_\theta(\tilde{i^p}, \tilde{i^n} \mid u)$, conditioned on u and parameters θ of the recommender $\mathbf{G}$.

Discriminative Adversary Training. The adversary $\mathbf{D}$ takes as input the synthetically generated item pairs $(\tilde{i^p}, \tilde{i^n})$ across all users, and an equal number of true pairs (i^p, i^n) sampled as described above. It performs two tasks:

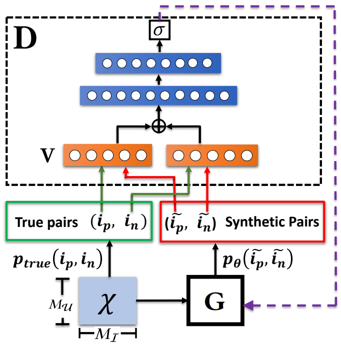
- $\mathbf{D}$ learns latent representations $\mathbf{V} = [\mathbf{v}_i, i \in \mathcal{I}]$ for the set of items with dimensionality d.
- Additionally, $\mathbf{D}$ learns a discriminator function $f_\phi(i^p, i^n)$ simultaneously with $\mathbf{V}$ to estimate the probability of a pair (i^p, i^n) being drawn from $p_{true}(i^p, i^n)$.

$$\mathbf{D}_\phi(i^p, i^n) = \sigma(f_\phi(i^p, i^n)) = \frac{1}{1 + \exp(-f_\phi(\mathbf{v}_{i^p}, \mathbf{v}_{i^n}))}$$

We implement $\mathbf{D}_\phi$ via two simple symmetric feedforward ladders followed by fully connected layers (Figure 2). With the parameters of $\mathbf{G}$ (*i.e.,* θ) fixed, ϕ and $\mathbf{V}$ are optimized by stochastic gradient methods to maximize the log-likelihood of the true pairs, while minimizing that of synthetic pairs with a balance parameter μ,

$$\phi^*, \mathbf{V}^* = \arg\max_{\phi} \sum_{u \in \mathcal{U}} \mathbb{E}_{(i^n, i^p) \sim p_{true}(i^p, i^n)} \left[\sigma(f_\phi(i^p, i^n))\right] + \mu.\mathbb{E}_{(\tilde{i^p}, \tilde{i^n}) \sim p_\theta(\tilde{i^p}, \tilde{i^n} \mid u)} \left[\log(1 - \sigma(f_\phi(\tilde{i^p}, \tilde{i^n})))\right] \tag{1}$$

Figure 2: Architecture details for the discriminative adversary D trained in tandem with base recommender G



Recommender Model Training. The more confident the distinction of the fake pairs generated as $(\tilde{i^p}, \tilde{i^n}) \sim p_\theta(\tilde{i^p}, \tilde{i^n} \mid u)$ by adversary $\mathbf{D}$, the stronger the penalty applied to $\mathbf{G}$. As previously described, synthetic pairs $(\tilde{i^p}, \tilde{i^n})$ are drawn as $\tilde{i^n} \propto f_\mathbf{G}(\tilde{i^n} \mid u, \mathcal{X})$, and $\tilde{i^p}$ randomly drawn from $\mathcal{X}_u^\mathcal{P}$. Thus,

$$p_\theta(\tilde{i^p}, \tilde{i^n} \mid u) \propto \frac{1}{|\mathcal{X}_u^\mathcal{P}|} f_\mathbf{G}(\tilde{i^n} \mid u, \mathcal{X}) \tag{2}$$

For sanity, we shrink $p_\theta(\tilde{i^p}, \tilde{i^n} \mid u)$ as $p_\theta(u)$ in the following equations. Our goal is to reinforce the associations of the niche items recommended by $\mathbf{G}$ to the popular items in user history. This is achieved when the synthetic pairs cannot be distinguished from the true ones, *i.e.*, $\mathbf{D}_\phi(\tilde{i^p}, \tilde{i^n})$ is maximized for the synthetic pairs sampled for each user. Thus, there are two terms in the recommender's loss, first the base objective $O_\mathbf{G}$ and second, the adversary term with weight λ. Note that $\mathbf{D}$'s parameters $\phi, \mathbf{V}$, are now held constant as $\mathbf{G}$ is optimized (alternating optimization schedule).

$$\begin{aligned}\theta^* &= \arg\max_\theta -O_\mathbf{G} + \lambda \sum_{u \in \mathcal{U}} \mathbb{E}_{(\tilde{i^p}, \tilde{i^n}) \sim p_\theta(u)} \left[\log D(\tilde{i^p}, \tilde{i^n})\right] \\ &= \arg\min_\theta O_\mathbf{G} + \lambda \sum_{u \in \mathcal{U}} \mathbb{E}_{(\tilde{i^p}, \tilde{i^n}) \sim p_\theta(u)} \left[\log(1 - D(\tilde{i^p}, \tilde{i^n}))\right]\end{aligned} \tag{3}$$

Since the second term (adversary) involves discrete item samples drawn on a per-user basis, it cannot be directly optimized by standard gradient descent algorithms. We thus apply policy gradient based reinforcement learning (REINFORCE) [12, 14] to approximate the gradient of the adversary term for optimization. Let us denote the gradient of the second term of eq. (3) for $u \in \mathcal{U}$ as $\nabla_\theta J^\mathbf{G}(u)$,

$$\begin{aligned}\nabla_\theta J^\mathbf{G}(u) &= \nabla_\theta \mathbb{E}_{(\tilde{i^p}, \tilde{i^n}) \sim p_\theta(u)} \left[\log(1 - D(\tilde{i^p}, \tilde{i^n}))\right] \\ &= \sum_{(\tilde{i^p}, \tilde{i^n}) \in \mathcal{I}^\mathcal{P} \times \mathcal{I}^\mathcal{N}} \nabla_\theta p_\theta(u) \log(1 + \exp(f_\phi(\tilde{i^p}, \tilde{i^n})) \\ &= \sum_{(\tilde{i^p}, \tilde{i^n}) \in \mathcal{I}^\mathcal{P} \times \mathcal{I}^\mathcal{N}} p_\theta(u) \nabla_\theta \log(p_\theta(u)) \log(1 + \exp(f_\phi(\tilde{i^p}, \tilde{i^n})) \\ &= \mathbb{E}_{(\tilde{i^p}, \tilde{i^n}) \sim p_\theta(u)} \left[\nabla_\theta \log(p_\theta(u)) \log(1 + \exp(f_\phi(\tilde{i^p}, \tilde{i^n}))\right] \\ &\approx \frac{1}{K} \sum_{k=1}^{K} \nabla_\theta \log(p_\theta(u)) \log(1 + \exp(f_\phi(\tilde{i^p}, \tilde{i^n}))\end{aligned} \tag{4}$$

The last step introduces a sampling approximation, drawing K sample-pairs from $p_\theta(u)$. Before adversarial training cycles, the recommender $\mathbf{G}$ can be pre-trained with loss $O_\mathbf{G}$, while $\mathbf{D}$ can be pre-trained with just the maximization term for true pairs. Our overall objective can be given by combining eq. (1), eq. (3),

$$\begin{aligned}O = \min_\theta \max_\phi O_\mathbf{G} + \lambda \sum_{u \in \mathcal{U}} \mathbb{E}_{(i^p, i^n) \sim p_{true}(i^p, i^n)} \left[\log D_\phi(i^p, i^n)\right] + \\ \mu . \mathbb{E}_{(\tilde{i^p}, \tilde{i^n}) \sim p_\theta(\tilde{i^p}, \tilde{i^n} \mid u)} \left[\log(1 - D_\phi(\tilde{i^p}, \tilde{i^n}))\right]\end{aligned}$$

On the whole, our framework employs a minimax strategy for iterative refinement: While the adversary progressively identifies finer distinctions between true and synthetic pairs thus refining the learned inter-item association structure, the recommender incorporates it in the item recommendations made to users.

4 EXPERIMENTS

In this paper, we employ a Variational Auto-Encoder (VAE-CF) [9] and Denoising Auto-Encoder (CDAE) [15] as our base recommender models $\mathbf{G}$. Results on the *ml-20m* dataset already indicate strong long-tail performance of stochastic VAE-CF (fig. 3) in comparison to deterministic CDAE [15]. Thus, performance gains in niche-item recall for VAE-CF with our adversarial training are particularly significant. We use two publicly available user-item datasets suitable for recommendation,

- **Movielens (*ml-20m*)**[1]: We binarized the available feedback matrix with a threshold of 5. Only users who watched atleast 10 movies were retained.
- **Ask-Ubuntu Stack Exchange**[2]: Tags were assigned to users if they Liked, Commented, Answered or asked a Question with the respective tags. Users with atleast 10 distinct tags were retained.

Similar to [9], we employ strong generalization with train, test, validation splits. Models are trained with training user interactions, while the interactions in the validation and test sets are split in two. One subset is fed as input to the trained model, while the other is used to evaluate the system output (ranked list) on $NDCG@100, Recall@K, K = 20, 50$. The architecture and training procedure is adopted from [9] for comparison. We set tradeoff parameter λ to multiple values and explore it's effect on recommendation over different sets of items, grouped by popularity. The balance parameter μ was set to 1 and $\mathbf{D}$ used a feed-forward network with 2 hidden layers (300, 100) as in fig. 2 (*tanh* activations and sigmoid output layer) and 300-dimensional embedding layers. All items with less than 0.5% appearance (`< 1 in 200`) were discarded, with negligible impact on results.

We will first analyze the composition of the `top 100` recommendations of $\mathbf{D} + \mathbf{G}$, against $\mathbf{G}$ trained in isolation. All items are split into four quartiles based on their popularity. We demonstrate the effect of the tradeoff λ on the `top 100` items for validation set users, by analyzing the quartiles they appear from (Table 1). Clearly, the recommendations from our model with higher values of λ improve the niche-tag coverage. But is this necessarily a good thing? Only if the overall performance is not degraded by poor recommendations. We analyze the overall recommendation performance against VAE-CF and CDAE in Table 2. Conventional baselines such as [4] are shown to be significantly weaker than our neural base recommender models.

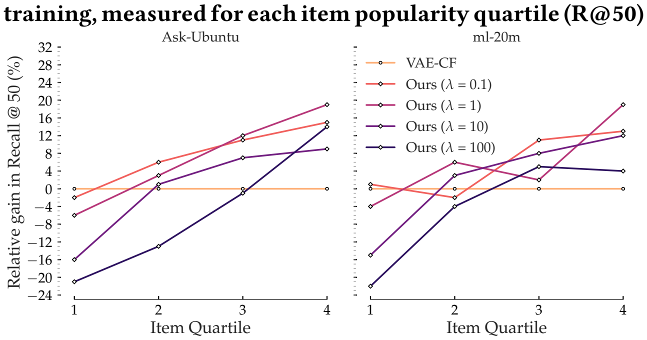
[1]https://grouplens.org/datasets/movielens/20m/
[2]https://archive.org/details/stackexchange

Table 1: Composition of top-100 item recommendations to users in item popularity quartiles (Q1-Most Popular Items)

Method	ml-20m				Ask-Ubuntu			
	Q-1	Q-2	Q-3	Q-4	Q-1	Q-2	Q-3	Q-4
CDAE (G_1)	74%	26%	0%	0%	97%	3%	0%	0%
D+G_1($\lambda = 0.1$)	61%	23%	10%	6%	76%	14%	7%	3%
D+G_1($\lambda = 1$)	62%	21%	11%	6%	73%	16%	6%	5%
D+G_1($\lambda = 10$)	61%	19%	12%	8%	65%	19%	11%	5%
VAE-CF (G_2)	64%	24%	8%	4%	60%	25%	9%	6%
D+G_2($\lambda = 0.1$)	58%	23%	12%	7%	53%	25%	12%	10%
D+G_2($\lambda = 1$)	59%	21%	13%	7%	55%	21%	13%	11%
D+G_2($\lambda = 10$)	59%	20%	13%	8%	54%	22%	14%	10%

Note that CDAE does not make *any* niche item recommendations (Q3 and Q4). Integrating our adversary to train CDAE results in a significant jump in long-tail coverage. To further dissect the above results, we will now observe our relative gains in *Recall*@50 compared to VAE-CF for each item quartile (Figure 3). We chose VAE-CF for comparison due to it's stronger long-tail performance.

Figure 3: Relative improvement over VAE-CF with adversary training, measured for each item popularity quartile (R@50)



As expected, our strongest gains are observed in Quartiles-3 and 4, which constitute long-tail items. Although there is a slight loss in popular item performance for $\lambda = 1$, this loss is not significant owing to the ease of recommending popular items with auxiliary models if required. We observe the values of tradeoff λ between 0.1 and 1 to generate balanced results.

We now analyze overall recommendation performance against VAE-CF and CDAE in Table 2 (**N** = NDCG, **R** = Recall). Even though our models recommend very different compositions of items (table 1), the results exhibit modest overall improvements for $\lambda = 0.1$ and $\lambda = 1$ over both the base recommenders. Clearly, the additional niche recommendations are coherent since there is no performance drop. However, larger λ values hurt the recommender performance. It is thus essential to balance the adversary objective and base recommender to obtain strong overall results.

Table 2: Overall recommender performance on ml-20m and Ask-Ubuntu datasets

Method	ml-20m			Ask-Ubuntu		
	N@100	R@20	R@50	N@100	R@20	R@50
CDAE (G_1)	0.34	0.27	0.37	0.29	0.30	0.46
VAE-CF (G_2)	0.51	0.44	0.57	0.42	0.45	0.59
D+G_2($\lambda = 0.1$)	0.53	0.45	0.59	0.43	0.46	0.61
D+G_2($\lambda = 1$)	0.52	0.44	0.58	0.42	0.46	0.59
D+G_2($\lambda = 10$)	0.48	0.41	0.55	0.40	0.43	0.56
D+G_2(λ=100)	0.42	0.37	0.51	0.38	0.41	0.53

5 CONCLUSION AND FUTURE WORK

In this paper, we investigated an adversarial learning framework to overcome sparsity in long-tail item recommendation. Our approach modernises conventional neighbor models, learning flexible associations guided by the feedback data. Our approach improved the long-tail performance of VAE-CF, which by itself outperforms CDAE by a significant margin. In future work, we plan several interesting directions. Integration of inter-user or cross associations with the item structure learned by the base recommender could prove valuable. Extension of our idea to retrieval problems to recover niche but relevant documents can prove impactful. Although our empirical results indicate reasonable model convergence, we plan to explore the Wasserstein metric [1] to provide a meaningful and smooth measure of the distance between the two competing distributions, with improved theoretical and empirical stability.

REFERENCES

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875* (2017).
[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.
[3] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.
[4] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. Ieee, 263–272.
[5] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009).
[6] Adit Krishnan, Ashish Sharma, and Hari Sundaram. 2017. Improving Latent User Models in Online Social Media. *arXiv preprint arXiv:1711.11124* (2017).
[7] Aaron Q Li, Amr Ahmed, Sujith Ravi, and Alexander J Smola. 2014. Reducing the sampling complexity of topic models. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 891–900.
[8] Xiaopeng Li and James She. 2017. Collaborative variational autoencoder for recommender systems. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 305–314.
[9] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. *arXiv preprint arXiv:1802.05814* (2018).
[10] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 287–296.
[11] Sanjay Purushotham, Yan Liu, and C-C Jay Kuo. 2012. Collaborative topic regression with social matrix factorization for recommendation systems. *arXiv preprint arXiv:1206.4684* (2012).
[12] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*. 1057–1063.
[13] Chong Wang and David M Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 448–456.
[14] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 515–524.
[15] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 153–162.
[16] Hongzhi Yin, Bin Cui, Jing Li, Junjie Yao, and Chen Chen. 2012. Challenging the long tail recommendation. *Proceedings of the VLDB Endowment* 5, 9 (2012), 896–907.