Adversarial Collaborative Neural Network for Robust Recommendation

Feng Yuan
University of New South Wales
Sydney, Australia
feng.yuan@student.unsw.edu.au

Lina Yao
University of New South Wales
Sydney, Australia
lina.yao@unsw.edu.au

Boualem Benatallah
University of New South Wales
Sydney, Australia
boualem.benatallah@gmail.com

ABSTRACT

Most of recent neural network(NN)-based recommendation techniques mainly focus on improving the overall performance, such as hit ratio for top-N recommendation, where the users' feedbacks are considered as the ground-truth. In real-world applications, those feedbacks are possibly contaminated by imperfect user behaviours, posing challenges on the design of robust recommendation methods. Some methods apply man-made noises on the input data to train the networks more effectively (e.g. the collaborative denoising auto-encoder). In this work, we propose a general adversarial training framework for NN-based recommendation models, improving both the model robustness and the overall performance. We apply our approach on the collaborative auto-encoder model, and show that the combination of adversarial training and NN-based models outperforms highly competitive state-of-the-art recommendation methods on three public datasets.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; *Information retrieval*; *Retrieval models and ranking*; • **Computing methodologies** → *Adversarial learning*;

KEYWORDS

deep learning, adversarial learning, item recommendation

ACM Reference Format:
Feng Yuan, Lina Yao, and Boualem Benatallah. 2019. Adversarial Collaborative Neural Network for Robust Recommendation. In *42nd Int'l ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR'19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3331184.3331321

1 INTRODUCTION

Most modern recommender systems consider the users' feedback as the ground-truth when making recommendation, which is probably not true in real life. For example, the explicit rating scores given by a user could be contaminated by his misoperations, resulting in an unpredictable amount of natural noise that affects the recommendation quality. In previous works, some methods manage to filter out noisy feedbacks from the data [1]. Others rely on the algorithm/model to deal with such feedbacks, e.g., adding extra bias terms in the prediction model [9, 14] and corrupting the input data so that the model is forced to learn robust parameters, known as the denoising auto-encoder (DAE) [10, 12]. In a recent work[7], the authors impose adversarial noises on the MF-based Bayesian personalized ranking (MF-BPR) [11] model, leading to an approach called the adversarial matrix factorization (AMF), to improve the model robustness.Different from the above works, in this paper, we focus on taking the advantage of deep neural networks combined with adversarial training. Thanks to the adoption of nonlinear transformation, there is no fundamental performance limitation as in the AMF approach. The main contributions are as follows:

- We propose a novel adversarial training framework for NN-based recommendation, based on which we design a minimax game for nonlinear model optimization. After thorough investigations on the impacts of adversarial noises on the model nonlinearity, we adopt our approach on the collaborative denoising auto-encoder (CDAE) model [12], based on which we propose two different models, the adversarial collaborative auto-encoder(ACAE) and an enhanced one, called fine-grained ACAE(FG-ACAE).
- We show salient improvements on performance and robustness by carrying out standard experiments on three different datasets, using both ACAE and FG-ACAE.

2 PROPOSED METHOD

In this section, we present the details of the proposed method. We define the adversarial noises as the model parameter perturbations maximizing the overall loss function [7]:

$$n_{adv} = \underset{\|N\| \le \epsilon}{\arg\max}\ loss(\Theta + N) \tag{1}$$

where ϵ controls the noise level, $\|\cdot\|$ denotes the L_2 norm , and Θ is the model parameters. Inspired by the fast gradient method in [2], the solution of Eq. (1) is approximated by:

$$n_{adv} = \epsilon \frac{\partial loss(\Theta + N)/\partial N}{\|\partial loss(\Theta + N)/\partial N\|} \tag{2}$$

The above noise is further served as the opponent factor in the minimax game of the training procedure. In general, the noise can be added in various ways, resulting in a generic loss function:

$$\underset{\Theta}{\arg\min}\ \underset{\|N_i\| \le \epsilon}{\arg\max}\ loss_{ORG}(\Theta) + \sum_{i=1}^{S} \lambda_i loss_{ORG}(\Theta + N_i) \tag{3}$$

where $loss_{ORG}$ denotes the loss function of the original model without adversarial training. Θ is the set of model parameters and S is the total number of ways to add the noises. λ_i controls the noise impact from the i-th source. N_i denotes the noise from source i.

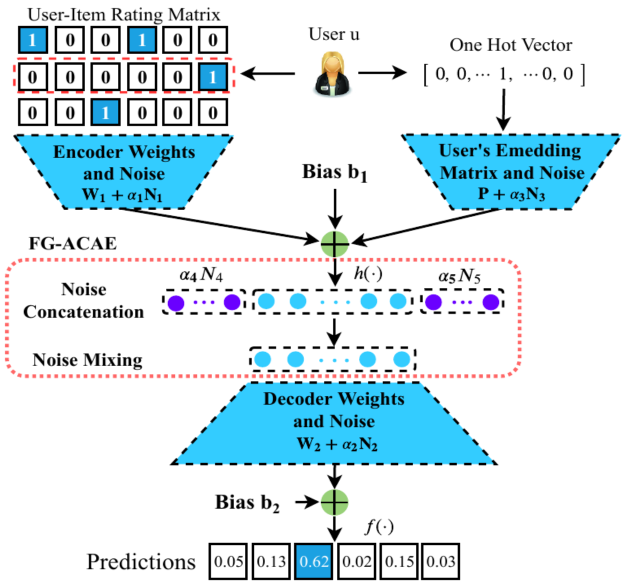

Figure 1: Adversarial Collaborative Neural Network

We then apply our framework on the modified CDAE model, where we remove the corruption on the input data and insert a noise mixing layer, resulting in a model that we call the collaborative auto-encoder (CAE). We note that the activation function for the mixing layer is set to identity. To decide how the noise should be inserted, initial experiments on the impacts of noise are carried out. For brevity, we only give our conclusions here. When added on the encoder or decoder weights, the adversarial noise poses more detrimental impacts. This can be explained by noticing the number of entries in the encoder or decoder weights is much larger than that of other positions, such as user embeddings **P**. A straightforward way is only adding the noise onto those positions that have the most significant impacts. Thus, by adding noise only on encoder/decoder weights and setting all other noise sources to zero, we arrive at the ACAE model [13]. However, ACAE only utilizes one parameter ϵ to control all noise sources, to benefit more from the adversarial training, we further insert noises in a fine-grained manner. Shown in figure 1, noise terms are controlled separately using different coefficients, so that for those positions which are less affected by the adversarial noise, a larger noise coefficient should be applied. In addition, we concatenate two additional noise vectors on the first hidden layer, after which a fully-connected layer is employed to mix the noise. We call this model fine-grained ACAE(FG-ACAE). We note that in FG-ACAE, ϵ is still in effect, as a general noise level control. We set $h(\mathbf{x}) = 1/(1+e^{-\mathbf{x}})$, $f(\mathbf{x}) = \mathbf{x}$, from which we obtain the optimal performance. We employ cross-entropy loss function $loss_{CE} = -y \log \sigma(\hat{y}) - (1-y) \log(1 - \sigma(\hat{y}))$ due to the binary nature of the input data. Inserting these conditions into Eq. (3), we have the following loss function:

$$loss = loss_{CE}(\mathbf{y}, \hat{\mathbf{y}}(\Theta)) + \sum_{i=1}^{S} \lambda_i \, loss_{CE}(\mathbf{y}, \hat{\mathbf{y}}(\Theta, \alpha_i \mathbf{N_i})) + \gamma \left(\|\mathbf{W_1}\|^2 + \|\mathbf{W_2}\|^2 + \|\mathbf{W_3}\|^2 + \|\mathbf{b_1}\|^2 + \|\mathbf{b_2}\|^2 + \|\mathbf{P}\|^2 \right) \quad (4)$$

where $\hat{\mathbf{y}}$ denotes the predicted ratings, α_i denotes the noise coefficient, N_i denotes the additive adversarial noise, λ_i controls the adversarial regularization strength, γ controls the L_2 regularization strength, $\mathbf{W_1}$,$\mathbf{W_2}$,$\mathbf{b_1}$,$\mathbf{b_2}$ are the encoder/decoder weights and biases, and **P** is the user embedding matrix. $\mathbf{W_3}$ is the weight matrix for the noise mixing layer. For ACAE, $S = 2$ and $\alpha_i = 1 (i = 1, 2)$, while for FG-ACAE, $S = 5$ and α_is are adjusted to achieve the best performance. Applying adversarial training procedure involves two main stages: (1) pre-training the modified CAE to get optimal Θ; (2) re-training the above model by iteratively maximizing (Eq. (1)) and minimizing the loss (Eq. (3)). Algorithm 1 gives the training procedure, where We employ mini-batch gradient descent for optimization. All weights are initialized using normal distribution with all biases set as zeros. The noise terms are initialized using zeros and updated according to Eq. (2).

Algorithm 1: Training algorithm for (FG-)ACAE.

```
Input: Training dataset: users $\mathcal{U}$, items $\mathcal{I}$, ratings $\mathcal{Y}$,
       regularizer coefficients $\lambda_i$s, noise coefficients $\alpha_i$s, $L_2$
       regularizer $\gamma$, general noise strength $\epsilon$, learning rate $\eta$,
       batch size $B$
Output: Model parameters: $\Theta$
// Pre-training
Initialize the parameter set $\Theta$ with normal distribution;
while Pre-training convergence condition is not met do
    Randomly draw a mini-batch $\mathcal{U}'$ with size $B$ from $\mathcal{U}$;
    Update $\Theta$ for each batch: $\Theta \longleftarrow \Theta - \eta \nabla_\Theta loss(\mathcal{U}'; \Theta)$;
end
// Adversarial training
Initialize the parameter set $\Theta$ with the pre-trained values;
while Adversarial training convergence condition is not met do
    Randomly draw a mini-batch $\mathcal{U}'$ with size $B$ from $\mathcal{U}$;
    $loss(\mathcal{U}'; \Theta) \longleftarrow Eq.(4)$ ;
    $\mathrm{N}_{i,adv} \longleftarrow Eq.(2)$;
    Update $\Theta, \eta$ with Adagrad;
end
return $\Theta$
```

3 EXPERIMENTS

3.1 Datasets

Table 1: Dataset Statistics

Dataset	User#	Item#	Ratings#	Density
CiaoDVD	17, 615	16, 121	72, 665	0.03%
MovieLens-1M	6, 040	3, 706	1, 000, 209	4.47%
FilmTrust	1, 508	2, 071	35, 497	1.14%

1. **Ciao** The CiaoDVDs dataset is collected by [3]. In this dataset, the users' feedbacks are given by ratings in the scale of $1-5$. Following [7], to binarize the data, we set ratings above 3 as 1 and others as 0. To deal with the repetitive ratings that are generated by the same user to the same item at different timestamps. We merge these ratings to the earliest timestamp.
2. **MovieLens-1M** The MovieLens dataset is widely used for recommender systems in both research and industry. We choose the 1M subset [5] which includes users' ratings ranging from 1

to 5. We use the same processing method as CiaoDVDs to get the binary data.

3. **FilmTrust** The FilmTrust dataset is crawled from the FilmTrust website by [4]. The ratings are given in the scale of $0.5 - 4$, so we set the ratings above 2 as 1 and others as 0 to binarize it.

3.2 Evaluation Metrics

We follow the procedure in [8] to generate the testing set and use the *leave-one-out* evaluation protocol. Specifically, for each user, we leave out the lastest user-item interaction and randomly select 200 unrated items into the testing set. For those datasets without timestamp information(i.e.,FilmTrust), we randomly select one interaction for each user to obtain the testing set. After training, we rank the predicted scores for testing items to generate a ranking list for each user. The N largest scored items are then used for performance evaluation. Hit Ratio (HR) is used to count the number of occurances of the testing item in the top-N ranking list while Normalized Discounted Cumulative Gain (NDCG) considers the ranking position of the item, where a higher position is assigned with a higher score. We average the metrics over all users and conduct one-sample paired t-test to judge the statistical significance when necessary.

3.3 Baselines

- **ItemPop** ItemPop is a non-personalized ranking method that generates the ranking list using the item popularity measured by the number of interactions in the training set.
- **MF-BPR**[11] This approach belongs to MF-based models with the BPR loss function, which is a competitive method for personalized item recommendation.
- **CDAE**[12] CDAE is based on DAE where the input data is corrupted by noise before fed into the neural network. We use the original code released by the authors and select proper activation and loss functions according to [12].
- **JRL**[15] is a NN-based recommendation model that uses a multi-layer perceptron (MLP) above the element-wise product of user and item embeddings. Despite its multi-view structure, we only adopt the ratings view for fair comparison.
- **NeuMF**[8] Neural MF is another NN-based model that combines the MF approach and MLP to extract latent interactions between users and items. We use the code from the author and follow the same pre-training procedure suggested in the original paper.
- **ConvNCF**[6] Convolutional Nueral Collaborative Filtering leverages CNN to learn correlations among user/item embedding dimensions. It has a much deeper structure than NeuMF and outperforms NeuMF on various datasets.
- **AMF**[7] Adversarial MF is a state-of-the-art approach that applies adversarial training on MF-BPR model with competitive results on various datasets. Using the original code provided by the authors and following the procedures in [7], we pre-train a MF-BPR model on which adversarial training is performed.

3.4 Results

The modified CAE is trained for 1000 epochs where it almost converges at epoch 500 for all datasets. We train it for another 500 epochs with adversarial noises added. On all datasets, the adversarial training stage converges for another 500 epochs in both ACAE and FG-ACAE cases. We tune the hyper-parameters carefully to obtain the best perfomance, where we set the batch size as 128 with learning rate selected from $\{0.001, 0.005, 0.01, 0.05, 0.1\}$. The noise level ϵ are tuned within $\{0.1, 0.2, 0.5, 1, 2, 5, 10, 15\}$ and regularizer coefficents λ_is and γ varying in $\{0.001, 0.01, \ldots, 100, 1000\}$. For FG-ACAE, α_is are tuned in the range of 0.01 to 100.

We evaluate the performance in terms of NDCG after each epoch with training traces shown in figure 2, where both ACAE and FG-ACAE achieve remarkable enhancement on all datasets. Particularly, FG-ACAE outperforms ACAE consistently, which is probably due to the increased amount of adversarial noise. In addition, with adversarial training applied, the training traces are often more unstable, because the noise acts as an opposite force against the minimization of the loss. However, this does not affect the overall performance improvement, especially for FG-ACAE.

The primary benefit of (FG-)ACAE is improving the CAE model's robustness against adversarial noises. To demonstrate this, we first adversarially trained the model under different noise strengths by varying ϵ. Then, on the trained model, we conduct robustness testing under various noise levels, i.e. ϵ ranging from 0 to 15. Figure 3 shows the robustness performance of ACAE and FG-ACAE on the MovieLens and FilmTrust datasets in terms of HR@5 under the adversarial noise level set at 8. We observe that in both datasets, when trained under a high noise level ($\epsilon = 7$), the robustness of the model against a certain level of adversarial noise is largely enhanced. This improvement is more obvious in the MovieLens dataset, as it is much denser than FilmTrust. At the same ϵ value, FG-ACAE consistently presents more robustness than ACAE due to its adding adversarial noises in more positions and controlling each noise level separately. Table 2 gives the comparison results of the baselines. On all datasets, we can see that FG-ACAE outperforms ACAE consistently. For the MovieLens dataset, ACAE outperforms all the baselines on the four chosen metrics. Especially, as adversarial training is more effective in the presence of nonlinear active functions, both ACAE and FG-ACAE exceed those of ConvMF which is the best baseline by 0.43% and 3.74% in terms of HR@5, respectively. We can observe similar results in the Ciao dataset. However, this is not true for the FilmTrust dataset. Although ACAE achieves a slightly higher HR, it fails to generate a better ranking position for the recommended items compared with AMF. With the enhanced FG-ACAE, it effortlessly outperforms AMF on all metrics. We also notice that the improvement over AMF on the MovieLens dataset is not only larger but also more significant than the FilmTrust dataset. One possible explanation lies in how sparse the dataset is. When the dataset is dense enough, despite its non-learning-to-rank nature, (FG-)ACAE can benefit more from the observed ratings, and generate better results for those missing ones to beat AMF. However, when the dataset is sparse, since AMF only samples a few unobserved user-item interactions to learn a ranking, it is less affected by the large amount of empty entries in the rating matrix, but (FG-)ACAE considers every empty entry, making it vulnerable to the noises imposed on those entries. For the Ciao dataset, although it is sparser than FilmTrust, yet it has much more users/items, enabling the model to see more distinctive input data, which leads to better performance.

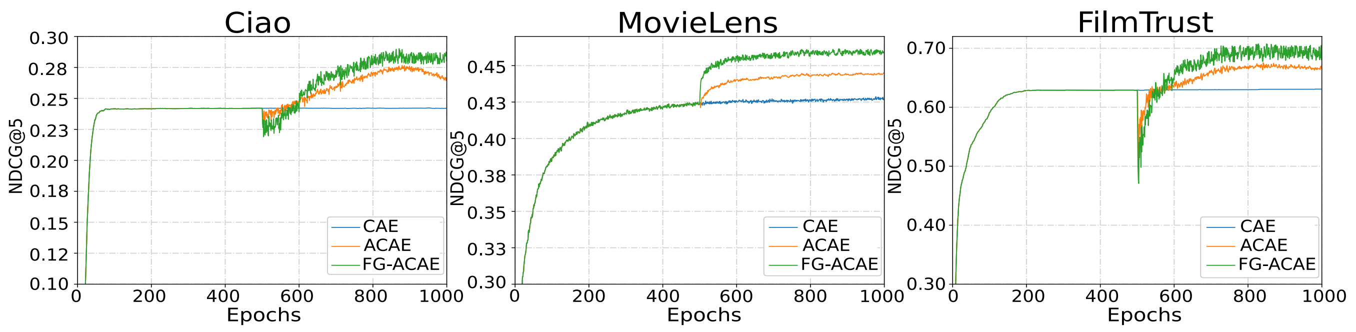

Figure 2: Adversarial Training Traces

Table 2: Comparison with Baselines. The best baselines are shown with asterisks and the best results are boldfaced.

	MovieLens-1M				Ciao				FilmTrust			
	HR@5	HR@10	NDCG@5	NDCG@10	HR@5	HR@10	NDCG@5	NDCG@10	HR@5	HR@10	NDCG@5	NDCG@10
ItemPop	0.310122	0.445847	0.212652	0.256204	0.242517	0.359323	0.182442	0.210237	0.656246	0.724774	0.531662	0.567708
MF-BPR	0.568163	0.716250	0.414792	0.462822	0.345351	0.456963	0.247260	0.283375	0.810294	0.861029	0.623186	0.640027
CDAE	0.574623	0.704986	0.428674	0.470944	0.343248	0.459516	0.242283	0.279717	0.816912	0.857353	0.631655	0.645153
JRL	0.572501	0.713687	0.430032	0.471640	0.364811	0.469962	0.261327	0.288609	0.835714	0.863573	0.667329	0.670354
NeuMF	0.583172	0.725037	0.430363	0.472680	0.370175	0.483051	0.265036	0.290573	0.830471	0.862710	0.675326	0.687054
ConvNCF	0.596314*	0.729088*	0.439655*	0.480397*	0.375427	0.487320	0.269314	0.296184	0.834637	0.870352	0.677435	0.696082
AMF	0.587543	0.726354	0.431428	0.476349	0.379751*	0.489710*	0.271169*	0.306817*	0.839706*	0.873088*	0.680039*	0.704567*
ACAE	0.598807	0.737949	0.444633	0.490474	0.381403	0.493466	0.274091	0.310169	0.843382	0.876471	0.677935	0.688810
FG-ACAE	**0.618631**	**0.750654**	**0.458644**	**0.513642**	**0.385972**	**0.503466**	**0.285054**	**0.322162**	**0.851693**	**0.880342**	**0.693980**	**0.712546**
Improve	3.74%	2.96%	4.32%	6.92%	1.64%	2.81%	5.12%	4.98%	1.43%	0.83%	2.05%	1.13%
Paired t-test	$p = 0.02$	$p = 0.01$	$p = 0.05$	$p = 0.04$	$p = 0.24$	$p = 0.20$	$p = 0.12$	$p = 0.16$	$p = 0.28$	$p = 0.32$	$p = 0.36$	$p = 0.30$

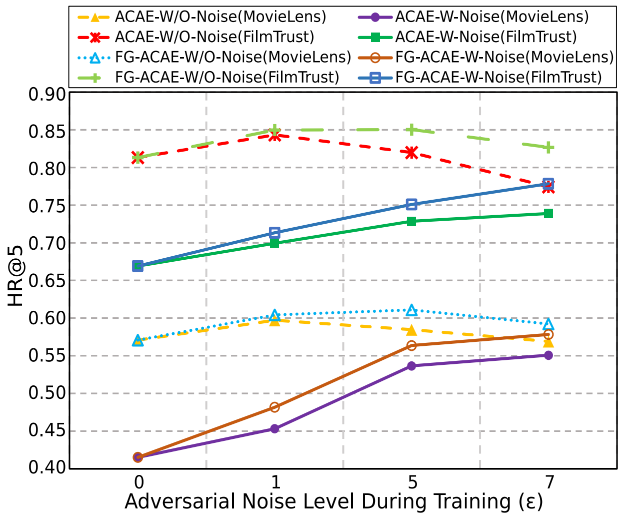

Figure 3: Degradation of HR@5 with Adversarial Noise ($\epsilon = 8$) on Decoder Weights After Adversarial Training

4 CONCLUSIONS

We propose a general adversarial training approach for NN-based personalized recommendation and develop two different models (ACAE and FG-ACAE) based on CDAE. We show that the adversarially trained models are superior to highly competitive state-of-the-art item recommendation methods. Furthermore, we improve the model's robustness against the adversarial noise. To the best of our knowledge, this work is the first in applying adversarial training techniques to NN-based recommendation models. In the future, we will further investigate the effect of adversarial training on deeper and more complicated NN-based recommendation models.

REFERENCES

[1] Chen-Yao Chung, Ping-Yu Hsu, and Shih-Hsiang Huang. 2013. βP: A novel approach to filter out malicious rating profiles from recommender systems. *Decision Support Systems* 55, 1 (2013), 314–325.
[2] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).
[3] Guibing Guo, Jie Zhang, Daniel Thalmann, and Neil Yorke-Smith. 2014. Etaf: An extended trust antecedents framework for trust prediction. In *Proceedings of the 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. 540–547.
[4] G. Guo, J. Zhang, and N. Yorke-Smith. 2013. A novel Bayesian similarity measure for recommender systems. In *IJCAI*. 2619–2625.
[5] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: history and context. *ACM Transactions on Interactive Intelligent Systems* 5, 4 (2016), 19.
[6] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer Product-based Neural Collaborative Filtering. In *IJCAI*. 2227–2233.
[7] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial personalized ranking for recommendation. In *SIGIR*. 355–364.
[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.
[9] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.
[10] Qibing Li and Xiaolin Zheng. 2017. Deep collaborative autoencoder for recommender systems: a unified framework for explicit and implicit feedback. *arXiv:1712.09043* (2017).
[11] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*. 452–461.
[12] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *WSDM*. 153–162.
[13] Feng Yuan, Lina Yao, and Boualem Benatallah. 2019. Adversarial Collaborative Auto-encoder for Top-N Recommendation. In *IJCNN*.
[14] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *Comput. Surveys* 52, 1 (2019).
[15] Yongfeng Zhang, Qingyao Ai, Xu Chen, and W Bruce Croft. 2017. Joint representation learning for top-n recommendation with heterogeneous information sources. In *CIKM*. 1449–1458.