On incorporating side information into linear recommender systems using adversarial training

Dora Jambor
dora.jambor@shopify.com
Shopify Inc.
Montreal, Quebec

Putra Manggala
putra.manggala@shopify.com
Shopify Inc.
Montreal, Quebec

ABSTRACT

The abundance of side information associated with items in e-commerce applications has provided numerous new possibilities for enriching recommender systems with rich sources of information. A variety of methods have shown positive effects of using side information on the recommendations' quality, often optimized with accuracy in mind.

In real-world applications, it can be desirable to incorporate different aspects of quality into recommendations such as domain knowledge, diversity, and fairness. A common challenge when optimizing for such qualities is to find an optimal trade-off behavior between accuracy and other metrics.

In this paper, we propose a new framework, AdRec, for incorporating side information into linear recommender systems via adversarial training. The adversarial training along with some proposed tunable hyperparameters allows practitioners to smoothly control the level of side information influence, and thereby achieve a desired trade-off profile. We evaluate AdRec on two datasets with manually generated side information to illustrate this method. We show how AdRec is generally more performant compared to popular baseline in terms of producing an optimal trade-off behavior. Lastly, we discuss some practical considerations for AdRec training.

CCS CONCEPTS

• **Information systems** → **Collaborative filtering**; **Personalization**; • **Human-centered computing** → *User models*;

KEYWORDS

Recommender Systems, Domain Knowledge, Adversarial Training

ACM Reference Format:
Dora Jambor and Putra Manggala. 2018. On incorporating side information into linear recommender systems using adversarial training. In *Proceedings of ACM*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Personalized top-N recommendations have been playing an important role in shaping the user experience in today's online world. Collaborative filtering (CF) is a standard approach for making recommendations based on user interactions [4] [16]. Most work on CF has focused on the *explicit feedback* setting where users express positive and negative preferences via ratings, likes or dislikes. In contrast, one-class collaborative filtering (OC-CF) deals with *implicit feedback* scenarios where the collected user interaction data represent only positive preferences, such as users' past purchases, clicks, views, installations, etc. Due to the vast availability of implicit signals in real-world systems, OC-CF is a natural choice in industry applications.

Besides implicit user-item interactions, there has been a great interest in incorporating additional information associated with the items, referred to as *side information*. Such can be domain knowledge from a multi-stakeholder environment, user perceptions captured by user research studies, product and marketing strategies or can also be product reviews, movie plots, *et cetera*.

The majority of the work on side information has focused on improving the quality of top-N recommenders as measured by precision. Such approaches include linear methods [14], matrix and tensor factorization [19], [9], and other hybrid methods [6]. Other than precision, there are a variety of benefits side information might offer in our recommenders. Such can be incorporating domain knowledge, mitigating cold start, and inducing diversity and fairness. In practice, optimizing for different quality metrics may come at the cost of lower precision. To our knowledge, this trade-off between item side information influence and precision has not been studied as extensively.

In this paper, we propose AdRec, an adversarial recommender that incorporates side information in a tunable manner via an adversarial training process. AdRec allows us to incorporate side information in a way that produces a desirable trade-off profile. As it builds upon state-of-the-art variants of Sparse Linear Methods (SLIM), it also retains the benefits of linear recommenders such as simplicity, interpretability, scalability and flexibility [18].

We conduct experiments to investigate how a trade-off between precision and a measure of side information influence emerges as we train AdRec, as compared to other baseline models, evaluated on two user interaction and side information datasets.

2 RELATED WORK

Various methods have been developed to incorporate side information in recommender systems. Most methods have focused on the rating prediction problem, whereas the top-N recommendation problem has received less attention.

Singh and Gordon [19] proposed the Collective Matrix Factorization method for movie rating prediction, where user-item purchases and item-feature content matrices are factorized into a common

ACM, RecSys, 2018

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

latent space such that the two types of information are leveraged via common item factors. They report improvements for rating predictions tasks.

Tensor factorization (TF) techniques with side information have also been explored. Lamba et al [9] uses a kernelized probabilistic model for TF to predict movie ratings. They report similar performance for accuracy as compared with other TF methods, however they report a lift in handling cold start problems.

Gunawardana et al [6] explores the trade-off one may incur when mitigating cold start problems with side information in recommenders. The authors proposes a hybrid approach to improve this trade-off by using a unified Boltzmann machines for top-N recommendation. They treat user-item profile information and side information as homogeneous features, where interaction coefficients between such features and user actions are learned in a coherent manner.

Lastly, Generative Adversarial Networks (GANs) [5] have not been extensively studied in recommender systems. To our knowledge, IRGAN [20] is the only method that builds on this framework. IRGAN uses adversarial training to enhance the document retrieval power of a generative information retrieval system. A discriminative model is used to discriminate between well-matched query-document tuples, from ill-matched ones. This acts like a guide for the generative model to recover the underlying relevance distribution over documents. Wang et al report improvements for several precision metrics in top-N recommendation tasks.

Due to the close relationship of AdRec with SLIM, the next section will describe these models in greater detail.

2.1 Definitions and Notations

Let $\mathcal{U}$ denote a set of users, and $\mathcal{I}$ a set of items, with $m = |\mathcal{U}|$ and $n = |\mathcal{I}|$. In OC-CF, we have a user-item matrix $\mathbf{R} \in \{0,1\}^{mxn}$, where $\mathbf{R}_{ui}$ represents the implicit interactions of user u with item i. We refer to $\mathbf{R}_{u:}$ as the partially observed user preference vector indicating the user's actions and inactions with a list of n items. For our later discussion on side information, let $\mathbf{A} \in \{0,1\}^{sxn}$ denote a matrix of item side information where s is the number of side information embeddings defined over n items. Let $\mathbf{A}_{e:}$ denote the individual embeddings.

In the top-N recommendation scenario [3], the goal is to produce a recommender matrix $\hat{\mathbf{R}} \in \mathbb{R}^{mxn}$ such that by sorting $\hat{\mathbf{R}}_{u:}$, we obtain the most relevant items for each user u.

2.2 SLIM: Sparse Linear Methods

SLIM [13] generates $\hat{\mathbf{R}}$ by learning an item-item similarity coefficients $\mathbf{W} \in \mathbb{R}^{nxn}$ such that $\mathbf{R} \sim \mathbf{RW}$. Formally, it learns $\mathbf{W}$ as the minimizer of the following regularized optimization problem:

$$\min_{\mathbf{W} \in C} \|\mathbf{R} - \mathbf{RW}\|_F^2 + \frac{\lambda}{2} \|\mathbf{W}\|_F^2 + \mu \|\mathbf{W}\|_1 \tag{1}$$

where $\lambda, \mu > 0$ are appropriate constants, and

$$C = \left\{ \mathbf{W} \in \mathbb{R}^{nxn} : diag(\mathbf{W}) = 0, \mathbf{W} \geq 0 \right\}. \tag{2}$$

Here, the last term of the objective function is the elementwise ℓ_1 norm of $\mathbf{W}$ so as to encourage sparsity in the coefficient matrix. The constraint $diag(\mathbf{W}) = 0$ prevents the trivial solution of $\mathbf{W} = \mathbf{I}_{nxn}$, while the nonnegativity constraint offers interpretability.

2.3 SLIM with side information

Various extensions of SLIM have been proposed in [14] to include side information about the items. In this paper, we discuss Collective SLIM (cSLIM), as this is one of the extensions that retains many of the desired properties of SLIM, and hence serves as an appropriate baseline in our evaluation.

cSLIM assumes correlation between users' co-interaction behaviors on two items and the similarity of the two items' intrinsic properties encoded in their side information $\mathbf{A}$. To enforce such correlation, in addition to $\mathbf{W}$ reproducing the user-item matrix $\mathbf{R}$, it must also satisfy $\mathbf{A} \sim \mathbf{AW}$. The original cSLIM methodology requires the same constraints as those in SLIM (2). Subsequent works [10] [17] [18] have however shown that removing the constraint (2) and $\ell 1$ regularization can not only simplify the training process, but can also yield more computationally efficient and relevant results. Due to the practicality of this finding for real-world recommenders, we define a relaxed cSLIM that solves the following objective:

$$\min_{\mathbf{W}} \|\mathbf{R} - \mathbf{RW}\|_F^2 + \beta * \|\mathbf{A} - \mathbf{AW}\|_F^2 + \frac{\lambda}{2} \|\mathbf{W}\|_F^2 \tag{3}$$

where $\|\mathbf{A} - \mathbf{AW}\|_F^2$ measures how well the item-item similarity model $\mathbf{W}$ fits the side information. The parameter β is used to control the relative influence of side information and the user-item matrix on $\mathbf{W}$. Aside from this added regularization term, cSLIM produces recommendations in the same way as SLIM does.

2.4 Generative Adversarial Networks

Before introducing our algorithm, we briefly review the prerequisite concepts underlying Generative Adversarial Networks (GAN).

GANs provide a framework to estimate generative models by simultaneously training two models: a generative model, G, and a discriminative model, D. The two models are pitted against one another in a two player game:

$$\min_G \max_D \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(x)}[log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_z(\mathbf{z})}[log(1 - D(G(\mathbf{z}))]. \tag{4}$$

G ingests a latent vector $\mathbf{z}$, sampled from some known prior $p_{\mathbf{z}}$, and produces $G(\mathbf{z})$, a sample of an implicit distribution p_{G}. While D is trained to assign correct labels to samples drawn from p_{G} versus p_{data}, G is optimized to minimize the probability of D classifying samples from p_{G} correctly.

Intuitively, D must learn the structure of the training data in order to effectively distinguish between real and synthesized samples. Contrarily, G must learn to mimic the training data in order to make D misclassify its generated output as a real sample.

3 ADVERSARIAL RECOMMENDERS

A similar intuition can be used to incorporate item side information into recommenders.

3.1 The AdRec Model

Formally, let p_{pref} be the distribution of user preferences and p_{data} be the distribution for side information embeddings. We denote our user-item training matrix as $\mathbf{R} \in \{0,1\}^{mxn}$, and our side information matrix as $\mathbf{A} \in \{0,1\}^{sxn}$. We define a linear recommender G: $\{0,1\}^{mxn} \rightarrow \mathbb{R}^{mxn}$ to be a function that takes a user preference vector $\mathbf{R}_{u:}$ and generates a user recommendation $\hat{\mathbf{R}}_{u:}$ which

follows the implicit distribution p_G. Let D be a second classifier $D\colon \mathbb{R}^{mxn} \rightarrow \mathbb{R}^{mx1}$ such that it outputs the probability that some input comes from p_{data} versus p_G. G and D then play the following two-player minimax game:

$$\min_G \max_D \mathbb{E}_{\mathbf{e}\sim p_{\text{data}}(\mathbf{e})}[logD(\mathbf{e})] \tag{5}$$

$$+ \gamma * \mathbb{E}_{\mathbf{r}\sim p_{\text{pref}}(\mathbf{r})}[log(\alpha - D(G(\mathbf{r})))] \tag{6}$$

$$+ \mathbb{E}_{\mathbf{r}\sim p_{\text{pref}}(\mathbf{r})}[\|\mathbf{r} - G(\mathbf{r})\|_2 + \lambda * \|\mathbf{W}_G\|_F^2], \tag{7}$$

where γ, α, and λ are appropriate constants and $\mathbf{W}_G$ is the parameter matrix of G.

In other words, maximizing (5) and (6) are such that D learns to discriminate between samples from $\hat{\mathbf{R}}$ and $\mathbf{A}$. Minimizing (7) will ensure that G reconstructs $\mathbf{R}$, similarly to (1), and finally, minimizing (6) will maximize D's probability of classifying the generated recommendations as side information.

3.1.1 Hyperparameters. Similarly to β in cSLIM, we introduce γ, to control the linear model's adversarial update 6, thus controlling the incorporation of side information. Note, when γ is non-zero, it behaves as a multiplier on the gradients of the generator's update. On the contrary, when it's zero, the adversarial objective is removed completely, and AdRec reduces to a simple linear recommender.

Second, as suggested in [15], we introduce one-sided label smoothing, denoted by α, to smooth the label for when an side information embedding is inputted to the discriminator. Smoothing labels prevents extreme extrapolation behavior in the discriminator. In our framework, lower α implies that less side information is incorporated to the produced recommendations.

Third, the number of epochs is useful for deciding when to stop training for a desired measure of side information.

4 PRELIMINARY EXPERIMENTS

We perform experiments on an internal private dataset where a training data is constructed from users' activations of features that could help enrich their experience. To reduce sparsity, users with less than four feature activations are removed. Additionally, we use a sub-sampled version of the Amazon review data dump released by [11] [12].

4.1 Side information

For the Amazon dataset, for each book, visual features are extracted from its product image, using the Caffe reference model [8] as per in [7]. For this experiment, we show how side information can be generated by a domain expert, by giving a plausible product strategy example using visual features. For each seed image in the set of lowest selling 100 items, we find 4 visually most similar items from the set of highest selling 200 items and encode this as an indicator vector of size n with the same ordering as a row from $\mathbf{R}$. A cell in this vector is one if it corresponds to either the seed image or the 4 most similar items. Visual similarity between two items is computed using Euclidean distance between their visual feature vectors. Side information matrix A is then obtained by stacking these 100 vectors to create a binary matrix of dimension 100 by n. Table 1 summarizes the statistics of these datasets.

Table 1: Summary of datasets used in evaluation.

Dataset	Instances	Items	nnz[1]
Internal dataset	3000	3437	70.860
Internal side information	3	3437	6
Amazon	4161	2000	38500
Amazon side information	100	2000	500

For our internal dataset, we craft a simple side information matrix by stacking three side information embeddings representing three pieces of domain knowledge. Here each row encodes information on which features a user with specific user criteria might find interesting.

4.2 Evaluation metrics

Our evaluation is done using **precision@k** (p@k) and **side information hit rate** (sihr@k), a new metric we introduce to measure AdRec's capability to incorporate side information, defined as:

$$\text{sihr@}k = \frac{1}{|\mathcal{U}||A|} \sum_{u\in\mathcal{U}, e\in A} \frac{|e \cap \{\hat{\mathbf{R}}_{u:}[:k]\}|}{k},$$

where A is the side information and $\hat{\mathbf{R}}_{u:}$ is user u's recommendations. This metric computes the average of the intersection between the user recommendation and side information row pair over all possible pairs. This measures how much the sets of items encoded in the side information embeddings end up in the top-N recommendations.

5 DISCUSSION

5.1 Internal dataset

Metrics are reported on the internal dataset using a random sample of 50% of users over 10 runs of both AdRec and cSLIM with a 95% confidence interval. The key takeaways can be summarized as follows:

- When $\gamma = 0.01$, AdRec yields a smooth trade-off profile across different values of α. As α increases, we demonstrate a trade-off via a decreasing p@k and an increasing sihr@k.
- This insight does not hold for cSLIM where the observed metrics weren't comparable to those for AdRec in their magnitude. Furthermore, different values of β did not allows us to control the incorporation of item side information, as proposed in the original paper.
- When γ, and β are 0, sihr@k for a k of 1 and 3 are 0, demonstrating that items specified in the side information datasets do not get ingested into top-N recommendations. As no side information is incorporated into the top-N recommendations, this has the highest p@k metrics for both algorithms as compared to other hyperparameter settings.

While table 2 shows the final trade-off metrics for training AdRec over 3 epochs and cSLIM over 5 epochs, in the case of AdRec, figure 1 provides more insight into the smooth trade-off between p@k and sihr@k as we train AdRec over 5 epochs for $\gamma \in \{0, 0.01\}$. From this graph we observe:

- When $\gamma = 0$, the model achieves the optimal p@1 by the second epoch.

- When $\gamma = 0.01$, we see that p@1 decreases and sihr@1 increases at different rates for different α values.

Given the difficulty of training adversarial models [1], as practitioners we find figure 2 helpful for choosing α and the number of epochs. For example, we would likely choose an α of 0.6 or 0.7, and 3 as the number of epochs, as these hyperparameter settings yield the best possible p@k and sihr@k trade-off, and we use p@k for $\gamma = 0$ as an upper bound.

5.2 Amazon dataset

This dataset is split into training/validation/testing by randomly sampling a purchase record for each user for validation and another one for testing. We then performed hyperparameter search using the validation set to identify sets of hyperparameters that could yield a trade-off behavior between precision and side information hit rate. We strive to obtain sets of hyperparameters that would yield this trade-off smoothly by varying just one hyperparameter. This is driven by practical considerations, as a practitioner could use this one degree of freedom to decide how much side information they would like to incorporate in the recommendations. We find that, given enough hyperparameter search, both relaxed cSLIM and AdRec (Table 3) were able to generate trade-off behaviors by just varying one hyperparameter.

Upon training AdRec for $\gamma \neq 0$ in this dataset, we find that varying the frequency of discriminator training with respect to generator training helps us discover the best hyperparameters in terms of precision and side information rate trade-off behavior in the validation set. This is due to the fact that in AdRec, optimizing for precision and optimizing for side information hit rate are two separate subprocesses during the training process. We can first train the generator until it is almost optimal, compute the precision with respect to the validation set, and then train the discriminator to incorporate side information. This is a more difficult process in cSLIM because there is no separation during training between the two optimizations.

Table 2: AdRec and cSLIM trade-off behavior for internal dataset.

γ	α	p@1 (%)	sihr@1 (%)	p@3 (%)	sihr@3 (%)
0.0	N/A	15.3 ± 0.24	0	9.3 ±0.23	0
0.01	0.6	14.4 ± 0.46	6.9 ± 0.18	8.1 ± 0.19	10.2 ± 0.06
0.01	0.7	13.3 ± 0.38	9.2 ± 0.15	6.7 ± 0.16	16.7 ± 0.07
0.01	0.8	10.3 ± 0.32	13.7 ± 0.17	6.3 ± 0.13	18.1 ± 0.10
0.01	0.9	11.1 ± 0.36	15.7 ± 0.15	6.7 ± 0.16	18.1 ± 0.11
β	N/A	p@1 (%)	sihr@1 (%)	p@3 (%)	sihr@3 (%)
0.0	...	2.7 ± 0.21	0	2.7 ±0.21	0
100	...	2.54 ± 0.19	2.77 ± 0.22	2.55 ±0.19	2.77 ± 0.22

The table shows the average precision estimates for 50% of users, over 10 runs with a 95% confidence interval. The above table corresponds to AdRec ran over 3 epochs, whereas the below table reports metrics for cSLIM, that ran for 5 epochs.

Table 3: AdRec trade-off behavior for Amazon dataset

γ	α	p@1 (%)	sihr@1 (%)
0.0	N/A	3.32 ± 0.24	2.10 ± 0.11
0.001	0.95	3.31 ± 0.21	2.32 ± 0.13
0.003	0.95	3.32 ± 0.22	3.12 ± 0.15
0.005	0.95	3.31 ± 0.20	3.41 ± 0.11
0.008	0.95	3.35 ± 0.10	3.56 ± 0.10
0.010	0.95	3.36 ± 0.12	3.89 ± 0.12
β	...	p@1 (%)	sihr@1 (%)
0.0	...	3.31 ± 0.21	2.10 ±0.11
0.05	...	3.32 ± 0.18	2.22 ± 0.08
0.08	...	3.20 ± 0.16	3.13 ± 0.12
0.1	...	3.23 ± 0.19	3.23 ± 0.09
0.2	...	3.19 ± 0.21	3.61 ± 0.12
0.3	...	3.20 ± 0.12	3.75 ± 0.11

The trade-off behavior between our two metrics indicates: as γ/β increases starting from 0, p@1 remains around the same value, while sihr@1 increases.

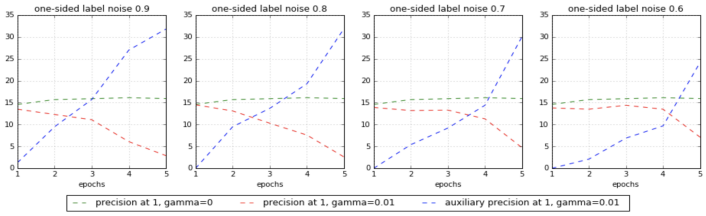

Figure 1: AdRec precision (%) trade-off over five epochs

6 CONCLUSION AND FUTURE WORK

We introduced AdRec, a framework designed for incorporating side information into linear recommender systems via adversarial training. Through careful experimentation and exhaustive hyperparameter search, we demonstrated the performance of AdRec compared to a variation of cSLIM, in terms of producing an optimal trade-off behavior between precision and side information hit rate. We have also provided some useful hyperparameters and heuristics for training AdRec via our case studies.

Currently this model is limited to side information that can be encoded as a set of indicator vectors with the same dimension as the user interactions as G is defined as an OC-CF recommender. In future work, we will explore the possibility of extending this beyond OC-CF problems so that richer side information can be encoded.

We will also explore ways to embed notions of diversity and fairness into our side information matrix, so that AdRec could be used to incorporate such notions into the recommendations. Additionally, we have opted to use a linear recommender due to its interpretability, however AdRec could also be formulated using higher capacity models such as in [2].

REFERENCES

[1] Martin Arjovsky and Léon Bottou. 2017. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862* (2017).

[2] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 191–198.

[3] Mukund Deshpande and George Karypis. 2004. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)* 22, 1 (2004), 143–177.

[4] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry. 1992. Using collaborative filtering to weave an information tapestry. *Commun. ACM* 35, 12 (1992), 61–70.

[5] 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.

[6] Asela Gunawardana and Christopher Meek. 2009. A unified approach to building hybrid recommender systems. In *Proceedings of the third ACM conference on Recommender systems*. ACM, 117–124.

[7] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback.. In *AAAI*. 144–150.

[8] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 675–678.

[9] Hemank Lamba, Vaishnavh Nagarajan, Kijung Shin, and Naji Shajarisales. 2016. Incorporating side information in tensor completion. In *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 65–66.

[10] Mark Levy and Kris Jack. 2013. Efficient top-n recommendation by linear regression. In *RecSys Large Scale Recommender Systems Workshop*.

[11] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 785–794.

[12] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 43–52.

[13] Xia Ning and George Karypis. 2011. Slim: Sparse linear methods for top-n recommender systems. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*. IEEE, 497–506.

[14] Xia Ning and George Karypis. 2012. Sparse linear methods with side information for top-n recommendations. In *Proceedings of the sixth ACM conference on Recommender systems*. ACM, 155–162.

[15] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*. 2234–2242.

[16] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. ACM, 285–295.

[17] Suvash Sedhain, Hung Hai Bui, Jaya Kawale, Nikos Vlassis, Branislav Kveton, Aditya Krishna Menon, Trung Bui, and Scott Sanner. 2016. Practical Linear Models for Large-Scale One-Class Collaborative Filtering.. In *IJCAI*. 3854–3860.

[18] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Darius Braziunas. 2016. On the Effectiveness of Linear Models for One-Class Collaborative Filtering.. In *AAAI*. 229–235.

[19] Ajit P Singh and Geoffrey J Gordon. 2008. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 650–658.

[20] 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.