# Convolutional Adversarial Latent Factor Model for Recommender System

Costa, Felipe Soares Da; Dolog, Peter

*Published in:*
Proceedings of the Thirty-Second International Florida Artificial Intelligence Research Society Conference, Sarasota, Florida, USA, May 19-22 2019.



*Publication date:*
2019

*Document Version*
Publisher's PDF, also known as Version of record

The Thirty-Second International Florida
Artificial Intelligence Research Society Conference (FLAIRS-32)

# Convolutional Adversarial Latent Factor Model for Recommender System

**Felipe Costa,**[1] **Peter Dolog**[1]
[1]Aalborg University
Selma Lagerløfs Vej 300
Aalborg 9220
{fcosta,dolog}@cs.aau.dk

## Abstract

The high accuracy of Top-$N$ recommendation task is a challenge in the systems with mainly implicit user feedback considered. Adversarial training has presented successful results in identifying real data distributions in various domains (e.g., image processing). Nonetheless, adversarial training applied to the recommendation is still challenged especially by interpretation of negative implicit feedback causing it to converge slowly as well as affecting its convergence stability. The researchers (He et al. 2018a; Chae et al. 2018) denotes the misinterpretation to high sparsity of the implicit feedback and discrete values characteristic from items recommendation. To face these challenges, we propose a novel model named convolutional adversarial latent factor model (CALF), which uses adversarial training in generative and discriminative models for implicit feedback recommendations. We assume that users prefer observed items over generated items and then apply the pairwise product to model the user-item interactions. Additionally, the hidden features become input data of our convolutional neural network (CNN) to learn correlations among embedding dimensions. Finally, Rao-Blackwell sampling is adopted to deal with the discrete values optimizing CALF and stabilizing the training step. We conducted extensive experiments on three different benchmark datasets, where our proposed model demonstrates its efficiency for item recommendation.

## Introduction

The internet has been facing information overload due to a large amount of shared data on the Web. Recommender systems proposes to overcome the information overload problem, aiming to predict user's preferences based on her/his history or popular items. Collaborative filtering (CF) has been the most commonly used method (Sarwar et al. 2001; Adomavicius and Tuzhilin 2005). Among the CF techniques, matrix factorization (MF) has become the most popular (Costa and Dolog 2018; Koren 2008) due to high accuracy in modeling the interaction between users and items such as browsing, rating, and clicking in latent space. Lately, implicit feedback has been extensively explored due to its practicality and accessibility in online services, turning the goal of recommender systems from rating predictions to learning to rank task. The recommendation model aims to predict a personalized ranking over a set of items for each user based on the similarities among the users and items. Nevertheless, unobserved items from implicit feedback lead us to misinterpret negative values because they may be unseen items or items whose user dislike.

To handle the research challenges mentioned above, we explore adversarial training to model users preferences from implicit feedback. Following the concept given by (Goodfellow, Shlens, and Szegedy 2014), Generative Adversarial Networks (GAN) have two components: a generative model trying to generate real samples and a discriminative model discriminating whether the samples are real or not. The idea is to train the model to defend against an adversary, such as the fake samples. Adversarial training has gained success in image processing and natural language processing, however in recommender systems it faces two issues: highly sparse data and discrete values. The sparse data may cause gradient vanishing or update instability, and the discrete values do not allow the adversarial training to directly optimizes using the gradient descent. Recently, (Wang et al. 2017) proposed Information Retrieval GAN (IRGAN) which applies adversarial training in the information retrieval field. IRGAN uses policy gradient strategy to obtain the model parameters. However, the variance of the estimated gradients increases linearly according to the number of items, making this solution impractical in recommender systems, since a large number of items may increase the vulnerability of adversarial training.

In order to solve the research challenges mentioned above, we propose a new adversarial training model for implicit recommender systems named CALF. Considering the adversarial training as a battle, the generative model aims to identify the user preferences by fighting with the discriminative model. The discriminative model aims to estimate the distribution distance between the generative model and the user preferences, while the generative model aims to minimize the estimated distance by capturing the actual distribution. Assuming the user prefers observed items over the generated items and the adversarial training as a battle, the generative and the discriminative models are oppo-

nents alternately optimizing the pairwise loss function. The goal is to improve the discriminator's judgment by minimizing the pairwise objective function, while the generator tries to generate user preferred items. The adversarial training process helps to handle the negative samples and avoid to design specific sampler as required by the policy gradient method. Moreover, we replace the non-differentiable item sampling by a differentiable item generating procedure using Rao-Blackwell sampling allowing the convolutional adversarial latent factor model (CALF) to update the gradients derived from the discriminator into the generator smoothly. Moreover, CALF is trained by the standard gradient descent method rather than policy gradient.

The paper presents the following contributions:

- A new model named CALF to improve the prediction based on user's preferences;
- An adversarial training model for a better learning correlation between the embedding dimensions and accelerating the convergence;
- A differentiable sampling method to deal with the discrete values allowing CALF to optimizes with gradient descent;
- Empirical evaluation in three benchmark datasets demonstrates the effectiveness of the CALF model.

## Related Works

Extensive work has been done in recommendations using explicit feedback. However, many online services rely on implicit feedback, where the main task from the recommendation system perspective is to provide a personalized list of items to each user rather than to predict the user ratings. Researchers from the recommender systems field have been investigating neural network techniques applied to collaborative filtering due to their ability to learn feature representations.

(He et al. 2017) introduced NCF to model user-item interaction function with implicit feedback combining a conventional MF with a multi-layer perceptron. Recently, an extended version named CNCF (He et al. 2018a) uses an interaction map layer applying the outer product to model pairwise correlations between embedding dimensions. Furthermore, the embedded vectors are used as input for CNN to learn the user-item interactions. Despite the effectiveness of NCF and CNCF, the neural collaborative filtering models have neglected adversarial perturbations causing vulnerabilities in their performance. Therefore, researchers proposed adversarial training for collaborative filtering, such as Adversarial Matrix Factorization (AMF) (He et al. 2018b), IRGAN (Wang et al. 2017)), and CFGAN (Chae et al. 2018). AMF applies adversarial personalized ranking (APR) on the MF method. IRGAN uses adversarial training into information retrieval field through element-wise product and defines the objective function via a probability-based method. CFGAN explores vector-wise adversarial training to solve the problem of discrete items. However, in both IRGAN and CFGAN the discriminative model performs as a binary classifier whose predicted values represent the probability that a user liked an item. In contrast, CALF is a pairwise method applying a generative and discriminative model based on CNN learning using adversarial training to learn user-item interactions.

## Problem Formulation

The research problem investigated in the paper is defined as follows: *How to improve Top-$N$ recommendation task using convolutional adversarial latent factor model?*

Consider a set of users $U = \{u_1, ..., u_M\}$ and a set of items $I = \{i_1, ..., i_N\}$. Let $Y = \{y_{vj} \in \mathbb{R}^{M \times N} | 1 \leq y_{vj} \leq 5\}$ denotes the rating matrix, where $y_{vj}$ is the rating of user $v$ on item $j$, and we label as $unk$ if it is unknown. Then, model the matrix $Y$ with implicit feedback as Eq. 1.

$$Y_{vj} = \begin{cases} 0, & \text{if } y_{vj} = unk \\ 1, & \text{otherwise} \end{cases} \tag{1}$$

Latent factor models define the recommender systems as the problem of predicting the preference score of each unobserved entry in $Y$ to further rank the list of items. We generate the scores as defined in Eq. 2.

$$\hat{Y}_{vj} = F(u_v, i_j | \Theta) \tag{2}$$

where $\hat{Y}_{vj}$ is the predicted score of interaction $Y_{vj}$ between user $u_v$ and item $i_j$, $\Theta$ is the model parameters, and $F$ is the function which estimates the predicted scores based on $\Theta$. The function $F$ leads to find the optimal list of items for an individual user according to users preferences.

Traditionally, MF methods define the function $F$ based on element-wise product of $p_v$ and $q_j$ to predict $\hat{Y}_{vj}$ as demonstrated by (Koren 2008), where $p_v$ and $q_j$ defines the hidden latent factors of $u_v$ and $i_j$, respectively.

$$\hat{Y}_{vj} = F(u_v, i_j | \Theta) = \mathbf{p}_v^T \mathbf{q}_j \tag{3}$$

We apply the pairwise product to calculate the interactions between users and items. The advantage in comparison with the element-wise product is a better representation model for non-linear interactions between users and items through a deep learning architecture. The loss function of the pairwise method follows the strategy given by (Rendle et al. 2009) where the difference between the items' ranking scores is given by:

$$L_{vjk} = \ln \sigma(\hat{\mathbf{y}}_{vj} - \hat{\mathbf{y}}_{vk}), \tag{4}$$

where $\hat{\mathbf{y}}$ is a ranking score, and a small loss represents high confidence that user $v$ prefers item $j$ over item $k$.

We use the following notations in further sections: $u$ denotes a user and $i$ denotes an item; $v$ and $j$ are indexes used to represent $u$ and $i$, respectively. $Y$ defines the user-item rating matrix mapped using Eq. 1, where $I^+$ is the observed interactions and $I^-$ unobserved interactions. Finally, $(v, j)$ denotes the $-th$ element from the matrix $Y$.

## Proposed Model

Figure 1 illustrates CALF's architecture, where the design of the prediction model defined in Eq. 2 can be observed. In the following section, we describe the detailed architecture of the CALF model.

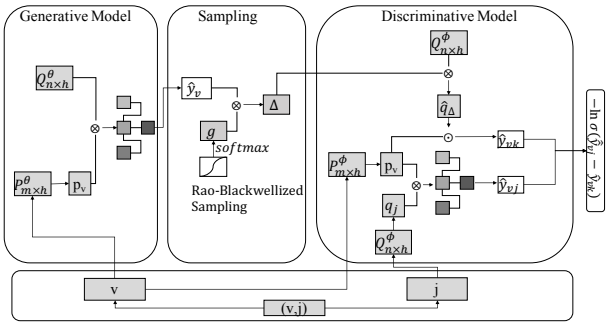


Figure 1: Convolutional Adversarial Latent Factor Model

## CALF Architecture

CALF has a generator $\mathbf{g}_\theta$ and a discriminator $\mathbf{d}_\phi$, where $\theta$ and $\phi$ are the parameters for the generative and discriminative models, respectively. Furthermore, $\mathbf{s}_\theta$ denotes the generator distributions and $\mathbf{s}_{real}$ denotes the user's true preferences. Specifically, the generator $\mathbf{g}_\theta$ tries to generate personalized items for each user through minimizing the distance between $\mathbf{s}_\theta$ and $\mathbf{s}_{real}$. While the discriminator $\mathbf{d}_\phi$ discriminates whether a user $v$ prefers item $j$ over $k$.

The convolutional layer in our generator and discriminator is inspired by (He et al. 2018a). The CNN is responsible for processing the useful signal from the pairwise user-item interactions. The embedding size of the input layer CNN is $64 \times 64$. The channel has 6 hidden layers where each of them has 32 feature maps. A feature map $\mathbf{y}$ in the hidden layer $l$ is represented as a $2D$ matrix of the interaction layer $\mathbf{S}^{ly}$. The stride is set as $[1, 2, 2, 1]$ which represents the *example*, *height*, *width*, and *channel*, respectively. The padding is defined as $SAME$. Considering these settings, the size of $\mathbf{S}^{lc}$ is half of its previous layer $l-1$. Let $\mathbf{y} = F_{(\mathbf{g}_\theta|\mathbf{d}_\phi)}$, where $F_{(\mathbf{g}_\theta|\mathbf{d}_\phi)}$ is the model function with the parameters $\theta$ and $\phi$. $\mathbf{y}$ is the mapping feature vector used to predict the final score.

Assuming that the user prefers the observed items over the generated items, the training step in the discriminator is a tuple $(v, j, k)$, where user $v \in U$, item $j \in I_v^+$ and item $k$ is sampled from $\mathbf{s}_\theta(k|v)$. The discriminator objective function is defined as:

$$J(\mathbf{g}_\theta, \mathbf{s}_\phi) = \max_\theta \min_\phi \sum_{v=1}^{m} \mathbb{E}_{j \sim \mathbf{s}_{real}(j|v) \&\& k \sim \mathbf{s}_\theta(k|v)} \ln \sigma(\hat{\mathbf{y}}_{vj} - \hat{\mathbf{y}}_{vk}), \tag{5}$$

where $\ln \sigma(\cdot)$ is the pairwise loss function. CALF adopts the logistic loss function proposed by (Rendle and Freudenthaler 2014), where the discriminator approximates the distance between $\mathbf{s}_\theta$ and $\mathbf{s}_{real}$. Note, while the discriminator in CALF minimizes the objective function and the generator maximizes the objective function, in the original GAN the loss function behaves oppositely.

## Sampling Strategy

Due to the discrete features (i.e., user ID, item ID, and other categorical variables) of the input data in recommender systems, the gradients derived from the objective function can not directly feed the generator. To solve this problem, IRGAN applies the policy gradient (reinforcement learning) to estimate the generator's gradients. However, the policy gradient presents two significant drawbacks: unstable training and slow convergence. To avoid these issues (Maddison, Mnih, and Teh 2016) propose to relax the discrete items. Rao-Blackwell sampling (Liu et al. 2018) proposes to reduce the variance of stochastic gradient estimators. In this paper, Rao-Blackwell sampling is adopted to solve the issues with the discrete items; then, we optimize CALF using gradient descent.

For user $v$, we denote $\hat{\mathbf{m}}_v \in \mathbb{R}^n$ as the the vector of item ranking scores and $\mathbf{g}_\Delta$ as the adversarial perturbations vector whose elements are randomly resulted from Rao-Blackwell estimator $(0, 1)$. The sampling is defined as:

$$\mathbf{\Delta} = \frac{1}{n} \sum_{i=1}^{n} \frac{2\hat{\mathbf{m}}_v + \mathbf{g}_\mathbf{\Delta}}{g_i + 1}, \tag{6}$$

where $\mathbf{\Delta}$ is the generated analogous one-hot item vector. To differentiate $\mathbf{\Delta}$ from the real items, we name it as a fake item. Each real item has a correspondent feature vector. However, it is not possible to define a feature vector for each fake item, due to existing an infinite number of fake items. Therefore, we define a differentiable method to obtain the feature vector of each fake item as:

$$\hat{\mathbf{q}} = \mathbf{\Delta}\mathbf{Q}, \tag{7}$$

where $\mathbf{\Delta} \in \mathbb{R}^n$ is a fake item, $\mathbf{Q} \in \mathbb{R}^{n \times h}$ is the feature vector of the items and $\hat{\mathbf{q}} \in \mathbb{R}^h$ is the latent vector of item $\mathbf{\Delta}$. $h$ is the number of hidden features. The described strategy proposes to overcome the discrete items challenge and facilitates the gradient information updates into the generator.

The parameters $\phi$ from the discriminator can now update via gradient descent aiming to minimize the objective function:

$$\phi \leftarrow \phi - lr \times \nabla_\phi \ln \sigma(\hat{\mathbf{y}}_\phi(v_j) - \hat{\mathbf{y}}_\phi(v_k)). \tag{8}$$

On the other hand, the parameters $\theta$ from the generator aim to maximize the objective function and optimizes using gradient ascent:

$$\theta \leftarrow \theta + lr \times \nabla_\theta \ln \sigma(\hat{\mathbf{y}}_\theta(v_j) - \hat{\mathbf{y}}_\theta(v_k)). \tag{9}$$

$lr$ denotes the learning rate. The proposed algorithm is described in Algorithm 1.

## Empirical Evaluation

We describe the experimental setup used to evaluate the CALF model performance explaining the datasets, evaluation metrics, baseline methods, and CALF settings. Moreover, we define the following research questions:

**Algorithm 1:** CALF Algorithm

**Input :** generator $\mathbf{g}_\theta$, discriminator $\mathbf{d}_\phi$, user-item interactions $Y$, learning rate $\eta$, number of epochs $epoch\_max$, and convergence criteria.

**Output:** top-n prediction from the prediction score $\hat{y}$.

initialize $\theta$ and $\phi$ randomly
epoch = 0
**while** *not converged* && *epoch* $<$ *epoch_max* **do**
  epoch+=1
  shuffle all observed interaction
  **foreach** *discriminator step* **do**
    **foreach** *observed feedback* $(v, j)$ *in current batch* **do**
      compute the generator items ranking scores $\hat{\mathbf{m}}_v$ for user $v$
      generate a fake item $k$ from $g_\theta$ for user $v$ via Equation 6
      get feature vector of fake item $k$ via Equation 7
      compute the pairwise loss $\ln \sigma(\hat{y}_{vj} - \hat{y}_{vk})$
      update the discriminator parameters $\phi$ via Equation 8
    **end**
  **end**
  **foreach** *generator step* **do**
    **foreach** *observed feedback* $(v, j)$ *in current batch* **do**
      compute the generator items ranking scores $\hat{\mathbf{m}}_v$ for user $v$
      generate a fake item $k$ from $g_\theta$ for user $v$ via Equation 6
      get feature vector of fake item $k$ via Equation 7
      compute the pairwise loss $\ln \sigma(\hat{y}_{vj} - \hat{y}_{vk})$
      update the discriminator parameters $\theta$ via Equation 9
    **end**
  **end**
**end**
**return** the top-$N$ items

**RQ1** Does the proposed model, CALF, outperform the state-of-art methods for item recommendations?

**RQ2** Does the Rao-Blackwell sampling strategy outperform the policy gradient method?

**RQ3** Does CALF improve the training convergence?

## Experimental Settings

**Datasets.** The experiments of the CALF model and baselines were conducted on three datasets: MovieLens 10M[1], Yelp [2], and Pinterest [3]. They are public benchmark datasets for recommender systems research community, and publicly available on their websites. The datasets were converted to implicit feedback following Eq. 1, where 1 denotes a user interaction with an item and 0 otherwise. Table 1 presents the statistics of the three datasets. We consider only users with minimum 20 interactions as recommended by (He et al. 2017) due to high sparsity in the datasets.

| Statistics | Movielens | Yelp | Pinterest |
|---|---|---|---|
| **# of Users** | 6,040 | 25,815 | 55,187 |
| **# of Items** | 3,706 | 25,677 | 52,400 |
| **# of Interactions** | 1,000,209 | 730,791 | 1,5000,809 |
| **Sparsity** | 95.53% | 99.89% | 99.73% |

Table 1: Statistics of the Datasets

**Evaluation Metrics.** To evaluate item recommendation using implicit feedback, we apply an adapted version of leave-one-out evaluation protocol (He et al. 2017; 2018a). The latest user-item interaction is held-out as the testing set, and the remaining interactions as the training set. We apply the strategy proposed by (Koren 2008) to minimize the time consumed to rank all items for every user during evaluation. Koren's strategy (Koren 2008) randomly samples 100 items that are not interacted by the user, and rank the test item among the 100 items. To evaluate the performance of item recommendation considering the top-$N$ task, we truncate the ranking list at position $N$. The metrics used to evaluate the ranking list are Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) (He et al. 2017). HR measures whether the testing item is in the top-$N$ list. NDCG considers not only the presence of an item in the top-$N$ list and its position.

**Baseline Methods.** The following state-of-art methods are used as baselines to compare the effectiveness of the CALF method.

- AMF (He et al. 2018b) applies adversarial personalized ranking (APR) on the shallow state-of-art MF method demonstrating good improvements in different datasets.
- CNCF (He et al. 2018a) applies the outer-product pairwise model to learn the correlations between the embedding dimensions, and CNN in their hidden layer to learn the correlations in hierarchical steps;
- IRGAN (Wang et al. 2017) applies the element-wise method to the generative and discriminative models, where the discriminator is a binary classifier, and it uses probability to obtain the objective function. Furthermore, IRGAN applies reinforcement learning to handle the discrete item problem.
- CFGAN (Chae et al. 2018) proposes a vector-wise adversarial training to deal with the discretization of items.

[1]https://grouplens.org/datasets/movielens/1m/

[2]https://github.com/hexiangnan/sigir16-eals

[3]https://sites.google.com/site/xueatalphabeta/academicprojects

| | **Movielens** | | | | Yelp | | | | Pinterest | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | HR@N | | NDCG@N | | HR@N | | NDCG@N | | HR@N | | NDCG@N | | RI |
| | N=5 | N=10 | N=5 | N=10 | N=5 | N=10 | N=5 | N=10 | N=5 | N=10 | N=5 | N=10 | |
| AMF | 0.5331 | 0.7255 | 0.3517 | 0.4444 | 0.1176 | 0.2385 | 0.0950 | 0.1065 | 0.7098 | 0.8972 | 0.4946 | 0.5658 | +31% |
| IRGAN | 0.5400 | 0.7301 | 0.3744 | 0.4665 | 0.1321 | 0.2550 | 0.1035 | 0.1113 | 0.7200 | 0.9002 | 0.5111 | 0.5832 | +25% |
| CNCF | 0.6103 | 0.8041 | 0.4316 | 0.5011 | 0.1578 | 0.2686 | 0.1073 | 0.1200 | 0.7489 | 0.9026 | 0.5367 | 0.5881 | +20% |
| CFGAN | 0.6805 | 0.8352 | 0.4991 | 0.5640 | 0.1829 | 0.2889 | 0.1184 | 0.1459 | 0.7668 | 0.9053 | 0.5513 | 0.5928 | +15% |
| CALF | **0.7124** | **0.8596** | **0.5121** | **0.6153** | **0.2037** | **0.3148** | **0.1364** | **0.1681** | **0.7811** | **0.9155** | **0.5742** | **0.6159** | - |

Table 2: Top-$N$ recommendation performance at $N = 5$ and $N = 10$. The bold font indicates the best results. RI indicates the relative improvement of CALF over the corresponding baseline on average.

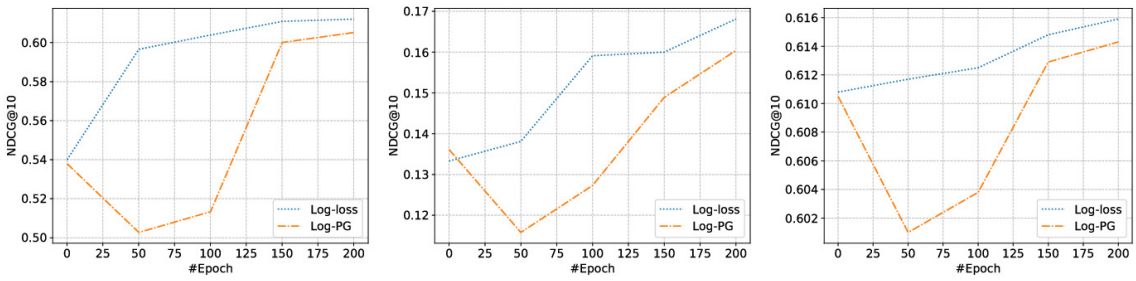


Figure 2: Differentiable sampling and policy gradient performance on Movielens (left), Yelp (center), and Pinterest (right).

**Modeling Settings.** We implement the CALF model in Python based on Tensorflow framework. CALF achieves the best performance in our experiments with the hyper-parameters set as below:

- The embedding size of 64 and optimized the loss function using mini-batch Adagrad with a batch size of 512;
- The learning rate $lr$ for both embedding and CNN is set as grid search from $\{0.001, 0.005, 0.01, 0.05, 0.1\}$;
- The adversarial regularization term $\lambda$ is set to 1 to ensure that the discriminator satisfies the Lipschitz constraint regarded by the logistic loss.

## Performance Comparison (RQ1)

Table 2 summarizes the results regarding the performance comparison for top-$N$ recommendation in the datasets. The analysis was made considering $N = 5$ and $N = 10$ as they are generally used to express the effectiveness of item recommendation.

CALF has an average relative improvement of 31% when compared to AMF. AMF applies adversarial perturbations to the shallow MF method, which may cause poor performance in comparison to other methods using CNN. Indicating that CNN has a better learning curve when representing users and items embeddings considering the item recommendation task.

IRGAN presents a good overall performance. However, due to its use of the element-wise product, CALF outperforms it with a relative improvement of 25%.

CNCF proves the advantage of applying pairwise correlations and using CNNs to learn non-linear user-item correlations. However, CNCF is not able to refine the relevance between user and items because it does not apply adversarial training. CALF presents a relative improvement of 20% in comparison to CNCF.

CALF outperforms CFGAN with a relative improvement of 15% due to its use of pairwise training and sample strategy, facilitating the adversarial training. Furthermore, CALF applies CNN, demonstrating a higher accuracy when learning non-linear user and item embeddings.

The results present the effectiveness of CALF, achieving the best overall performance in all datasets.

## Sampling Strategy Effectiveness(RQ2)

The sampling strategy for discrete values was evaluated considering CALF using logistic loss function with the policy gradient. The hyper-parameters from logistic loss and policy gradient have the same values for a fair comparison. We calculate the gradient of the policy strategy for user $v$ as follows:

$$
\begin{aligned}
\bigtriangledown_\theta J(\mathbf{g}_\theta, \mathbf{s}_\phi) &= \bigtriangledown_\theta \mathbb{E}_{j\sim \mathbf{s}_{real}(j|v) \&\& k\sim \mathbf{s}_\theta(k|v)} \\
&\qquad \ln \sigma(\hat{\mathbf{y}}_{vj} - \hat{\mathbf{y}}_{vk}) \\
&= \sum_{j=1}^{n}\sum_{k=1}^{n} \mathbf{s}_{real}(j|v) \bigtriangledown_\theta \left[\mathbf{s}_\theta(k|v) \ln \sigma(\hat{\mathbf{y}}_{vj} - \hat{\mathbf{y}}_{vk})\right] \\
&= \sum_{j=1}^{n}\sum_{k=1}^{n} \mathbf{s}_{real}(j|v)\mathbf{s}_\theta(k|v) \bigtriangledown_\theta \left[\log \mathbf{s}_\theta(k|v) \right. \\
&\qquad \left. \ln \sigma(\hat{\mathbf{y}}_{vj} - \hat{\mathbf{y}}_{vk})\right] \\
&= \mathbb{E}_{j\sim \mathbf{s}_{real}(j|v) \&\& k\sim \mathbf{s}_\theta(k|v)} \bigtriangledown_\theta \left[\log \mathbf{s}_\theta(k|v) \right. \\
&\qquad \left. \ln \sigma(\hat{\mathbf{y}}_{vj} - \hat{\mathbf{y}}_{vk})\right] \\
&\simeq \frac{1}{N|I_v^+|} \sum_{j=1}^{|I_v^+|}\sum_{k=1}^{N} \bigtriangledown_\theta \log \mathbf{s}_\theta(k|v) \ln \sigma(\hat{\mathbf{y}}_{vj} - \hat{\mathbf{y}}_{vk})
\end{aligned}
\tag{10}
$$

| | Movielens | | | | | Yelp | | | | | Pinterest | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **CALF** | **AMF** | **CNCF** | **IRGAN** | **CFGAN** | **CALF** | **AMF** | **CNCF** | **IRGAN** | **CFGAN** | **CALF** | **AMF** | **CNCF** | **IRGAN** | **CFGAN** |
| **D** | 1 m | - | - | 45 s | 50 s | 1.7 m | - | - | 55 s | 1.3 m | 1.9 m | - | - | 1.9 m | 1.9 m |
| **G** | 1.7 m | - | - | 1.5 m | 1.2 m | 3.9 m | - | - | 1.7 m | 2.5 m | 3.7 m | - | - | 4.9 m | 3.9 |
| **EC** | 50 | 60 | 50 | 60 | 60 | 50 | 100 | 90 | 120 | 100 | 70 | 90 | 80 | 70 | 70 |
| **TC** | 3 h | 4 h | 3.5 h | 5 h | 4.6 h | 4.6 h | 5.3 h | 5 h | 7 h | 6.5 h | 6.5 h | 6.9 | 6.8 h | 8 h | 7 h |

Table 3: Convergence time. - denotes the methods without generative and discriminative models; D is the discriminative model; G is the generative model; EC denotes the epoch convergence; and TC denotes the time convergence

where $N$ is the number of policy gradient sampling items.

Figure 2 illustrates the learning curve of logistic loss with gradient descent is stable in comparison with the policy gradient. Analyzing Figure 2, a peak is observed in the policy gradient loss when the adversarial training starts, but after some epochs, it drops, and, finally starts increasing again. On the other hand, the logistic loss using the sampling strategy for discrete values keeps stable during the training step. In other words, the policy gradient has slower convergence, which may be caused by the high variance of the policy gradient. Reducing the gradient variance improves adversarial training.

### Time Complexity Analysis (RQ3)

The complexity analysis of GAN models is $O(|Y|nh)$, where $|Y|$ denotes the number of user-item interactions. CALF has additional computations when compared, for example, with the policy gradient applied by IRGAN. However, the computational time in adversarial models relies on the iterations. Moreover, the generative and discriminative models of CALF alternates the optimization in each step, while in the other GANs the training epoch of the discriminator spends double time than the generator.

Table 3 presents the computational time spent by CALF in comparison with other models. Analyzing Table 3 CALF spends more time to train the generator and discriminator in each epoch. However, the convergence time is shorter compared to the other models. Therefore, considering the total training time, the sampling strategy adopted by CALF improves the computational time and the stability.

## Conclusion

We propose a convolutional adversarial latent factor model for items recommendations using implicit feedback, named CALF. A detailed description explains how CALF uses adversarial training for its recommendation. Furthermore, we presented the results of the conducted experiments.

CALF has proved to be useful for top-$N$ items recommendations considering implicit feedback. Moreover, learning deep representations for pairwise interactions among user and item embeddings improved the accuracy for predicting the user's preference score, as observed in the results of our experiments in the three benchmark datasets.

In the future, we will conduct investigations regarding richer contexts such as social relations and user's reviews. Moreover, we would like to apply attention mechanisms to learn user and item similarities.

## Acknowledgments

The authors wish to acknowledge the financial support and the fellow scholarship given to this research from the Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (grant# 206065/2014-0).

## References

Adomavicius, G., and Tuzhilin, A. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *TKDE* 734–749.

Chae, D.-K.; Kang, J.-S.; Kim, S.-W.; and Lee, J.-T. 2018. Cfgan: A generic collaborative filtering framework based on generative adversarial networks. CIKM'18, 137–146.

Costa, F., and Dolog, P. 2018. Hybrid learning model with barzilai-borwein optimization for context-aware recommendations. FLAIRS'18, 456–461.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *CoRR*.

He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; and Chua, T.-S. 2017. Neural collaborative filtering. WWW'17, 173–182.

He, X.; Du, X.; Wang, X.; Tian, F.; Tang, J.; and Chua, T.-S. 2018a. Outer product-based neural collaborative filtering. IJCAI'18, 2227–2233.

He, X.; He, Z.; Du, X.; and Chua, T.-S. 2018b. Adversarial personalized ranking for recommendation. SIGIR, 355–364.

Koren, Y. 2008. Factorization meets the neighborhood: A multifaceted collaborative filtering model. KDD, 426–434.

Liu, R.; Regier, J.; Tripuraneni, N.; Jordan, M. I.; and McAuliffe, J. 2018. Rao-blackwellized stochastic gradients for discrete distributions. *CoRR*.

Maddison, C. J.; Mnih, A.; and Teh, Y. W. 2016. The concrete distribution: A continuous relaxation of discrete random variables. *CoRR*.

Rendle, S., and Freudenthaler, C. 2014. Improving pairwise learning for item recommendation from implicit feedback. WSDM'14, 273–282.

Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. Bpr: Bayesian personalized ranking from implicit feedback. UAI'09, 452–461.

Sarwar, B.; Karypis, G.; Konstan, J.; and Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. WWW'01, 285–295.

Wang, J.; Yu, L.; Zhang, W.; Gong, Y.; Xu, Y.; Wang, B.; Zhang, P.; and Zhang, D. 2017. Irgan: A minimax game for unifying generative and discriminative information retrieval models. SIGIR'17, 515–524.