Collaborative Deep Denoising Autoencoder Framework for Recommendations

Jinjin Zhao†
Amazon.com
Seattle, WA
jinjzhao@amazon.com

Lei Wang
Alibaba Group
Hangzhou, China
wanglbox@gmail.com

Dong Xiang
Amazon.com
Seattle, WA
dongxian@amazon.com

Brett Johanson
Amazon.com
Seattle, WA
bjohanso@amazon.com

ABSTRACT

This work proposes a Collaborative Deep Denoising Autoencoder (CDDAE) framework for recommender systems to improve recommendation experiences. In this framework, a non-linear matrix factorization is computed through deep denoising autoencoder leveraging user-item interactions and auxiliary profile information. Different from existing matrix factorization and deep learning approaches, this framework combines autoencoder, collaborative filtering and embedded auxiliary information together to provide better recommendation experiences. Experiments conducted show CDDAE improves recommendation experiences by 5.6% in precision and 7.4% in recall averagely.

CCS CONCEPTS

• Personalization • Search and Recommendation

KEYWORDS

Search, recommendation, interaction, user behavior

1 Introduction

1.1 Overview

Memory-based Collaborative Filtering (CF) considers user-item interactions for predictions which assumes two users behaving (e.g., actions, clicks, views, likes.) similarly in the past will also do so in the future. GroupLens [1] from GroupLens lab, one of the first labs who studied automated recommender systems, is a successful and early generation of a memory-based CF system. However, memory-based CF is not scalable and fast enough in real-time systems. Model-based CF is investigated. The well-known model-based CF technologies [2][3][4][5] are Bayesian Belief Nets (BNs), Markov Decision Process (MDP) based CF approaches. Compared with memory-based CF which directly stores user-item interaction information in memory to make prediction without training in advance, model-based CF builds models based on user-item interactions to make predictions instead which benefits both latency and scalability. Since both memory-based CF and model-based CF can not incorporate with auxiliary information, content-based CF [6] is introduced to make predictions by leveraging item and user features. While content-based CF systems do not necessarily incorporate information in preference similarity across individuals [7], hybrid approaches taking in both user and content context into one system has been explored. Van den Oord [8] builds two separate learning stages utilizing memory-based or model-based CF and content-based CF to produce recommendations.

Recently, deep learning-based recommender systems have been actively investigated [9][10], where each user and item features are combined (or averaged, concatenated) to make predictions by following with several layers perceptrons. Deep structured semantic models [11] look into users textual behaviors (search queries, browsing histories) and textual content, then maps the users and items into a latent space where the similarity between users and their preferred items is maximized. There is also research to combine collaborative filtering and deep learning into collaborative deep learning based recommendation in recent years [12][13][14][15]. Wang [12] proposed a hierarchical Bayesian model called collaborative deep learning (CDL) which performs deep representation learning for the content information and collaborative filtering for the ratings (feedback) matrix. Autoencoder [16][17][18][19] is an approach recently introduced into the recommender system where non-linear matrix factorization is computed by the autoencoder framework with user-item ratings.

2 Proposed Methodology

In this work, the authors introduce Collaborative Deep Denoising Autoencoder (CDDAE) framework to jointly consider user profile, item profile and user-item interactions to make recommendations. This model builds the latent correlations from user-item preference as base knowledge, user profile and item profile as auxiliary knowledge which are separately embedded. A non-linear matrix factorization is performed with deep denoising autoencoder for the user-item ratings and embedded auxiliary features. Auxiliary knowledge can be incorporated in a flexible manner through CDDAE framework.

The contributions in this work are as described below:

1. Propose a Collaborative Deep Denoising Autoencoder (CDDAE) framework for general recommender systems,

SIGIR'19, July, 2019, Paris, France

where the shared latent space is constructed by encoding and decoding process.

2. Propose a flexible embedding structure to incorporate auxiliary user profile and item profiles, where all user-item ratings, user, and item profiles share the same latent space.
3. Conduct experiments to study the performance of different CF algorithms in both public dataset and real business dataset, where effectiveness and robustness of CDDAE are proved.

2.1 Autoencoder

Autoencoders were first introduced by Kramer [20] in 1991, which utilized Feed-Forward Neural Networks (FFNN) to learn a representation of input with a reduced dimension. Output in the network aims at reconstructing the input. The network is then trained by back-propagating the loss (e.g., squared error loss) during the reconstruction, where two parts are as below,

- Encoder $\varphi : x \rightarrow z$
- Decoder $\psi : z \rightarrow x$

Where φ, $\psi = \arg min_{\varphi,\psi} ||x - (\varphi * \psi)x||^2$. In the simplest case, there is only one hidden layer, where the encoder takes input *x* and maps it to *z*, then the decoder maps *z* into reconstruction *x*,

- Encoder: $z = \sigma(Wx+b)$
- Decoder: $x = \sigma(W'z+b')$

Where σ is a non-linear activation function, $x \in R^d$ is the input, $z \in R^k$ is the hidden node, $W \in R^{d*k}$ is weight matrix mapping input to hidden node, $W' \in R^{k*d}$ is the weight matrix mapping hidden node to reconstruction node, $b \in R^k$, $b' \in R^d$ as bias vectors.

2.2 Denoising Autoencoder

The Denoising Autoencoder (DAE) was first introduced by [21] in order to discover more robust features through autoencoding in addition to learning the identity function. DAE uses input x, is corrupted as $\tilde{x}$, and the network is trained to denoise and reconstruct input x. Common corruption choices include but are not limited to the additive Gaussian noise and multiplicative mask-out/drop-out noise. In this paper, the original corruption mechanism is utilized which randomly masks entries of the input by making them zero.

2.3 Collaborative Deep Denoising Autoencoder

The Collaborative Denoising Autoencoder (CDAE) [24] method is represented as one-hidden-layer neural network, where a latent vector for user V_u is encoded to achieve better recommendations compared to standard DAE methods. CDAE first corrupts input as $\tilde{x}_u$ with mask-out/drop-out approach. Then it maps corrupted input $\tilde{x}_u$ to a latent representation z_u as shown below in Equation 1.

$$z_u = h(W^T \tilde{x}_u + V_u + b) \quad (1)$$

In the output layer, z_u is mapped back to original input space to reconstruct the input vector,

$$\hat{x}_u = f(W'^T z_u + b') \quad (2)$$

The loss function, reconstruction error, is minimized as parameters learned within the Stochastic Gradient Descent (SGD) algorithm.

$$\arg\min \frac{1}{U} \sum_{u=1}^{U} E_{p(\hat{x}|x)} \mathrm{L}(\hat{x} - x) + R \quad (3)$$

Where R is the regression term with L2 norm,

$$R = \frac{\lambda}{2}(||W||_2^2 + ||W'||_2^2 + ||V||_2^2 + ||b||_2^2 + ||b'||_2^2) \quad (4)$$

In this paper, a framework is introduced which jointly combining DAE with a deep neural network, additional feature embeddings, and a collaborative filtering approach to model the hidden correlations between users and items.

Deep DAE. As shown in the CDDAE framework in **Figure 1**, the encoding process of deep DAE takes corrupted user-item rating $\tilde{R}_{u,i}$, jointly maps them into a bottleneck layer with user embedded features E_u, and item embedded features E_i. The bottleneck layer H is then the latent variable layer where user profile and item profile share the same latent space.

$$H = encoder(W^T * \tilde{R}_{u,i} + W_u{}^T E_u + W_i{}^T E_i + b) \quad (5)$$

Where W, W_u, and W_i are the corresponding weight matrices, b is the offset vector, and the *encoder* is a mapping function. In the decoding process, $R_{u,i}$, E_u , and E_i are reconstructed simultaneously with the deep neural network with pre-defined layer structure.

$$[R_{u,i}, E_u, E_i] = decoder(W'^T H + b') \quad (6)$$

In Equation 6 above, W' and b' are the corresponding weight matrix and offset vector, and the *Decoder* is a mapping function. In contrast to DAE, which is a one-hidden-layer neural network, both encoder and decoder are designed as a deep neural network to extract high order non-linear features in this work.

Feature Embedding. The item profile contains non-semantic structured data (e.g., author, popularity, date_created) and semantic unstructured data (e.g., topic, title, body, comments). Similarly, user profile contains non-semantic structured data (e.g., occupation, professional level) and semantic unstructured data (e.g., interests, focus). Both item and user profiles are separately embedded into a lower dimension as item_embedding vector E_i and user_embedding vector E_u with text embedding and one-hot embedding approaches.

Collaborative Filtering. Collaborative learning is achieved by minimizing reconstruction loss during the encoding and decoding process. The sigmoid function is used as the output layer activation function and binary cross-entropy as the loss function which has proved to be the best choice in solving binary classification problems.

$$f(s_i) = \frac{1}{1+e^{s_i}} \quad (7)$$

$$CE = -\sum_{i=1}^{C'=2} t_i \log(s_i) \quad (8)$$

In Equation 7 and 8, t_i is ground truth and s_i is the predicted score from the neural network output. C'=2 is the class number. $i \in (0,1)$ is the index of the class. Similarly, softmax as output layer activation function and categorical cross entropy as loss function is also experimented in following sections.

$$f(s_i) = \frac{e^{s_i}}{\sum_j^C e^{s_j}} \quad (9)$$

$$CE = -\log\left(\frac{e^{s_i}}{\sum_j^C e^{s_j}}\right) \quad (10)$$

Where C=2 as class number.

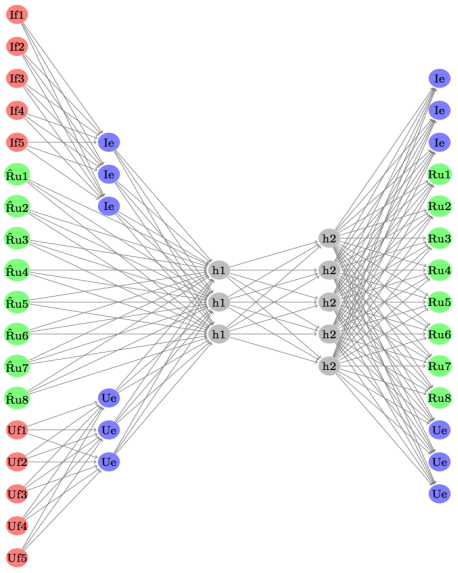

Figure 1: CDDAE framework, If and Uf are for item features and user features. Ie and Ue are for item embedding and user embedding. $\widehat{Ru}$ is the corrupted user-item rating. Ru is the reconstructed user-item rating. $h1$ and $h2$ are hidden layers.

3 Experimental Design and Results

3.1 Dataset Description

The authors experimented on the public dataset MovieLens-100K to test CDDAE's effectiveness and robustness as compared to the existing approaches CDAE and DAE. MovieLens-100K is a common dataset for evaluating performance of recommender systems.

MovieLens-100K dataset. This dataset contains 100K ratings from 943 users on 1682 movies. Each rating represents the user-item interaction, which is an integer ranging from 1 to 5. Ratings with 1 to 3 stars are transformed to zeros (0's) and the 4 and 5's are transformed into 1's. This is a widely used pre-processing approach on recommendation with implicit feedback[22][23]. User profile with user id, age, gender, occupation are also extracted as auxiliary information.

By reconstructing the interaction and personal profile, latent space with a much lower dimension is constructed and modeled, and the latent variables are used to make predications. The task of the two datasets are the same which is to recommend next Top N items for each user to achieve better recommendation experiences.

3.2 Evaluation Metrics

Top-N recommendation metrics are used to evaluate the outcomes of different approaches. Top-N items for each user are predicted with the highest scores which are assumed not adopted in the training dataset. Then predicted results are compared with the ground truth to check if the recommended items have been adopted by the user in the test dataset.

Precision and Recall. For each user, given a Top-N recommendation list $L_{N,pred}$ with N items, compared to ground truth list as L_{gt}, precision and recall is defined as,

$$P@N = \frac{L_{N,pred} \cap L_{gt}}{N} \quad (11)$$

$$R@N = \frac{L_{N,pred} \cap L_{gt}}{L_{gt}} \quad (12)$$

Considering all users, modifications to the precision calculation were used in a simplified way to reflect all users,

$$PU@N = \frac{\sum_u \min(L_{N,pred} \cap L_{gt}, 1)}{U} \quad (13)$$

where the number of users is counted who have at least one recommendation predicted correctly in the Top-N recommendations. $\min(L_{N,pred} \cap L_{gt}, 1) = 0$ reflects the user gets zero recommendation which he is interested, $\min(L_{N,pred} \cap L_{gt}, 1) = 1$ means the user at least gets one recommendation correctly predicted. Accumulated Intersection is also calculated over Union (IoU) as recall,

$$RU@N = \frac{\sum_u L_{N,pred} \cap L_{gt}}{\sum_u L_{gt}} \quad (14)$$

where $\boldsymbol{L_{N,pred} \cap L_{gt}}$ is the intersection over Top-N recommended and ground truth.

3.3 Experimental Design and Hyperparameters

On the public dataset, detailed experiments were conducted to compare CDDAE with existing the-state-of-art approaches DAE and CDAE with same hyperparameters, user-item rating dataset, and the evaluation metric defined in 3.2. Additionally, auxiliary user information is incorporated to prove the effectiveness and flexibility of CDDAE framework. Experiments with different combinations of hyper parameters are also conducted to evaluate its robustness.

Hyperparameters are experimented with corruption rate, regularizer penalties, output layer activation function, loss function, deep neural nets width and depth design, which are detailed in each

experiment below. In all experiments, Adam[24] algorithm is set as the optimizer, 0.001 is set as learning rate, 'relu' function is set as the hidden layers' activation function.

3.4 Experimental Results

In experiments below, comparable experiments are firstly conducted for CDDAE, CDAE and DAE to evaluate CDDAE's effectiveness. Secondly, experiments with various combinations of hyperparameters are conducted to evaluate CDDAE's robustness. Thirdly, experiment with auxiliary user information are conducted to prove that the flexible framework can incorporate additional information during constructing the latent space.

Experiment 1. First, comparable experiments are conducted with DAE and CDAE with the same hyper parameters listed below which are the default setting for CDAE. Comparisons between CDAE and other CF approaches are explored in [25], but is out of the scope of this experiment.

1. corruption rate (cr) : 0.5
2. regularizer penalty (rp) : 0.01
3. output layer activation function: 'sigmoid'
4. loss function: 'mean absolute error'
5. one hidden layer as bottleneck layer with width: 50

User-item rating data is declared in 3.1 where each user is represented with a fixed unique index. No auxiliary information is adopted in this experiment and DAE works by reconstructing user-item ratings through denoising, encoding and decoding process regardless user index. CDAE differentiates from DAE by adding in an additional user node to the input layer in the encoding process and then reconstructs user-item ratings through decoding process. The user node is represented by its index and CDDAE reconstructs both user-item ratings and the user node via denoising auto-encoding.

Figure 2 is the Top N precision and recall for all three approaches. N ranges from 1 to 10. **Table 1** lists the numbers in detail. The results show CDDAE performs better than DAE and CDAE by 3.1% and 3.1%, respectively, in the Top 1 recommendation for both precision and recall, 6.96% and 8.96% respectively in the Top 10 recommendations in precision, 10.1% and 13.4% respectively in Top 10 recommendations in recall. As the results indicate, CDAE performs worse as defined in [25] than DAE, which indicates that adding user node with an index into the encoding process might corrupt latent space by reducing the degree of freedom. However, reconstructing the user node in CDDAE helps organize latent representation better than DAE.

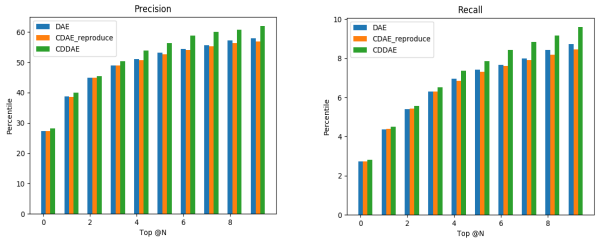

Figure 2: Top N Precision[eq.13]/Recall[eq.14] on MovieLens-100K with user-item rating information.

Table 1: Top N Precision[eq.13]/Recall[eq.14] on MovieLens-100K with user-item rating information.

Top @N(percentile)	1	2	3	4	5	6	7	8	9	10
DAE@Precision	27.36	38.71	44.96	48.89	51.01	53.13	54.40	55.67	57.16	57.90
CDAE@Precision	27.36	38.49	44.86	48.99	50.69	52.60	53.98	55.36	56.31	56.84
CDDAE@Precision	28.21	39.98	45.49	50.37	53.87	56.42	58.75	60.02	60.76	61.93
CDDAE/DAE	103.10%	103.29%	101.18%	103.04%	105.61%	106.19%	107.99%	107.81%	106.31%	106.96%
CDDAE/CDAE	103.10%	103.86%	101.42%	102.81%	106.28%	107.26%	108.84%	108.43%	107.91%	108.96%
DAE@Recall	2.74	4.36	5.41	6.30	6.96	7.42	7.67	8.01	8.42	8.72
CDAE@Recall	2.74	4.39	5.43	6.30	6.85	7.32	7.61	7.91	8.20	8.46
CDDAE@Recall	2.82	4.50	5.58	6.53	7.36	7.85	8.42	8.84	9.17	9.60
CDDAE/DAE	103.10%	103.16%	103.14%	103.70%	105.79%	105.71%	109.82%	110.46%	108.94%	110.10%
CDDAE/CDAE	103.10%	102.42%	102.73%	103.70%	107.43%	107.25%	110.58%	111.80%	111.90%	113.41%

From **Experiments 2 to 4**, auxiliary information are added to experiments to evaluate the effectiveness of CDDAE's flexible framework. Both embedded user profile and item profile can be adopted in this framework. 'user gender' as user profile is adopted in this experiment. Different hyperparameters are conducted to prove its robustness. CDAE results in **Experiment 1** are also compared with all CDDAE results as below.

Experiment 2. In this experiment, output activation function and loss function are adjusted with all other hyper parameters fixed as shown in **Table 2**. The results of the study are shown in **Figure 3** where CDDAE performs significantly better than CDAE by incorporating additional 'user gender' information. The results show the best performer from precision and recall as metrics is the combination of sigmoid and binary cross-entropy, which aligns with the existing research result about effectiveness test on the activation function and the loss function for binary classification problems. The results show mean absolute error (MAE) which measures Euclidean distance is a worst choice in solving binary classification problems.

Table 2. Hyper parameter design (output activation function, loss function) for Experiment 2

output activation function	loss function	cr	rp	layer depth	bottleneck layer width
Sigmoid	Binary Cross-Entropy	0.1	0.01	1	50
Sigmoid	Categorical Cross-Entropy	0.1	0.01	1	50
Sigmoid	Mean Absolute Error	0.1	0.01	1	50
Softmax	Binary Cross-Entropy	0.1	0.01	1	50
Softmax	Categorical Cross-Entropy	0.1	0.01	1	50
Softmax	Mean Absolute Error	0.1	0.01	1	50

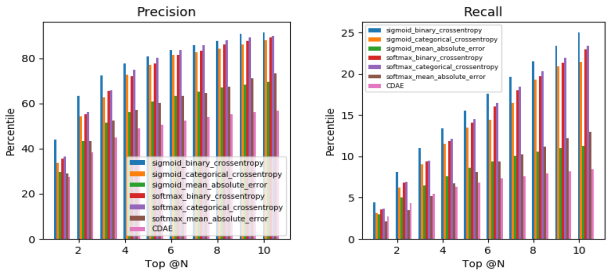

Figure 3: Top N Precision[eq.13]/Recall[eq.14] for CDDAE with activation functions and loss function. CDAE as baseline.

Experiment 3. The corruption rate and the regularizer penalty are experimented with the other parameters fixed in **Table 3**. The dataset in **Experiment 3** remains the same as in **2**, both parameters are designed to handle uncertainty and noise, the corruption rate reflects the denoising capability in reconstruction process, and the regularizer penalty impacts the capability in dealing with ill-posed problems and overfitting. The result are summarized in **Figure 4** below which shows that CDDAE achieves 5.9% precision improvement and 7.4% recall improvement compared to CDAE. It works slightly better around 0.7% improvement with higher corruption rate or higher regularizer penalty, which indicates a more robust latent space is constructed to prevent overfitting and to better deal with noise.

Table 3. Hyper parameter design (corruption rate, regularizer penalty) for Experiment 3

output activation function	loss function	cr	rp	layer depth	bottleneck layer width
Sigmoid	Binary Cross-Entropy	0.1	0.01	1	50
Sigmoid	Binary Cross-Entropy	0.3	0.01	1	50
Sigmoid	Binary Cross-Entropy	0.5	0.01	1	50
Sigmoid	Binary Cross-Entropy	0.7	0.01	1	50
Sigmoid	Binary Cross-Entropy	0.1	0.001	1	50
Sigmoid	Binary Cross-Entropy	0.3	0.001	1	50
Sigmoid	Binary Cross-Entropy	0.5	0.001	1	50
Sigmoid	Binary Cross-Entropy	0.7	0.001	1	50

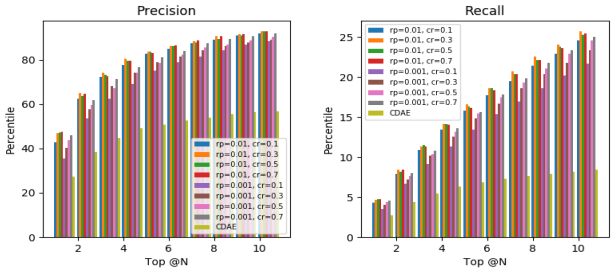

Figure 4: Top N Precision[eq.13]/Recall[eq.14] for CDDAE with corruption rate (cr) and regularizer penalty (rp). CDAE as baseline.

Experiment 4. In this experiment, experiments are conducted with different layer designs in the encoding and decoding process and the dataset remains the same in **Experiment 2** and **3**. First, layer depth is limited to 1, where the corrupted input layer is directly mapped to the bottleneck layer directly and the output layer is reconstructed directly from bottleneck layer. First, bottleneck layer width was varied, second, the layer depth was increased to 2, where an additional dense layer is added to model higher order features either in the encoding process or decoding process. Experiments with layer depth larger than 2 were not experimented within this research due to the MovieLens-100K dataset being too small to optimize millions of variables, which could result in local optimal. A detailed layer design is shown in **Table 4**.

Table 4. Hyper parameter design (layer width and depth) for Experiment 4

output activation function	loss function	cr	rp	layer depth	bottleneck layer width	additional layer width
Sigmoid	Binary Cross-Entropy	0.1	0.01	1	10	-
Sigmoid	Binary Cross-Entropy	0.1	0.01	1	20	-
Sigmoid	Binary Cross-Entropy	0.1	0.01	1	50	-
Sigmoid	Binary Cross-Entropy	0.1	0.01	1	100	-
Sigmoid	Binary Cross-Entropy	0.1	0.01	1	200	-
Sigmoid	Binary Cross-Entropy	0.1	0.01	1	400	-
Sigmoid	Binary Cross-Entropy	0.1	0.01	2	50	100 (1 layer in encoder)
Sigmoid	Binary Cross-Entropy	0.1	0.01	2	50	200 (1 layer in encoder)
Sigmoid	Binary Cross-Entropy	0.1	0.01	2	50	400 (1 layer in encoder)
Sigmoid	Binary Cross-Entropy	0.1	0.01	2	50	100 (1 layer in decoder)
Sigmoid	Binary Cross-Entropy	0.1	0.01	2	50	200 (1 layer in decoder)
Sigmoid	Binary Cross-Entropy	0.1	0.01	2	50	400 (1 layer in decoder)

In **Figure 5,** as layer depth is set to 1 and the bottleneck layer width increases from 10 to 50, both precision and recall imporve, however performance drops as the bottleneck layer width continually increases to 400. This indicates the width, or lack thereof, of a bottleneck layer limits the capability to describe the complexity of the latent space, and as the width of bottleneck layer increases, the degree of freedom result in a local optimal. In this experiment, it was determined the best bottleneck width is figured as 50 when layer depth is 1 and will consist to this setting in all following experiments. **Figure 6** shows an additional layer in the decoding process improves the performance, while adding one additional layer in encoding process does not improve performance at all. This indicates a deeper decoding process can help to extract higher order features and the width of additional layer ranging from 100 to 400 is not sensitive to the performance whether on the encoder or decoder side.

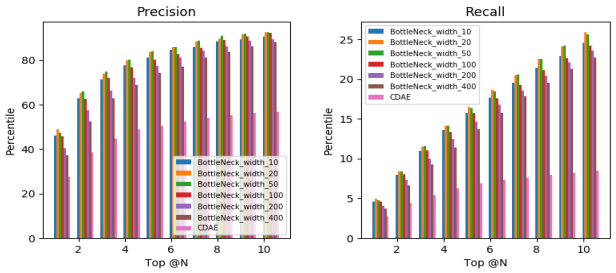

Figure 5: Top N Precision[eq.13]/Recall[eq.14] for CDDAE with layer width design. CDAE as baseline.

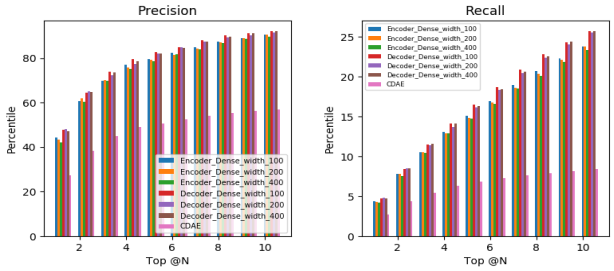

Figure 6: Top N Precision[eq.13]/Recall[eq.14] for CDDAE with layer depth design. CDAE as baseline.

4 Conclusion

In this work, a framework called Collaborative Deep Denoising autoencoder for recommender systems is proposed which computes a non-linear matrix factorization with auxiliary information incorporated flexibly through deep denoising autoencoder. User profile, item profile and user-item preference are mapped simultaneously into the same latent space. Experiment result shows that this approach outperforms 5.6% in precision and 7.4% in recall averagely on public dataset MovieLens-100K.

REFERENCES

[1] Su, Xiaoyuan, and Taghi M. Khoshgoftaar. "A survey of collaborative filtering techniques." Advances in artificial intelligence 2009 (2009).

[2] J. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Pro- ceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI '98)*, 1998.

[3] X. Su and T. M. Khoshgoftaar, "Collaborative filtering for multi-class data using belief nets algorithms," in *Proceedings of the International Conference on Tools with Artificial Intelli- gence (ICTAI '06)*, pp. 497–504, 2006.

[4] K. Miyahara and M. J. Pazzani, "Collaborative filtering with the simple Bayesian classifier," in *Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence*, pp. 679–689, 2000.

[5] G. Shani, D. Heckerman, and R. I. Brafman, "An MDP-based recommender system," *Journal of Machine Learning Research*, vol. 6, pp. 1265–1295, 2005.

[6] Balabanović, Marko, and Yoav Shoham. "Fab: content-based, collaborative recommendation." *Communications of the ACM*40.3 (1997): 66-72.

[7] A. Ansari, S. Essegaier, and R. Kohli, "Internet recommenda- tion systems," *Journal of Marketing Research*, vol. 37, no. 3, pp. 363–375, 2000.

[8] Van den Oord, Aaron, Sander Dieleman, and Benjamin Schrauwen. "Deep content-based music recommendation." In Advances in neural information processing systems, pp. 2643-2651. 2013.

[9] Covington, Paul, Jay Adams, and Emre Sargin. "Deep neural networks for youtube recommendations." In Proceedings of the 10th ACM conference on recommender systems, pp. 191-198. ACM, 2016

[10] Zhang, Shuai, Lina Yao, Aixin Sun, and Yi Tay. "Deep learning based recommender system: A survey and new perspectives." ACM Computing Surveys (CSUR) 52, no. 1 (2019): 5.

[11] Huang, Po-Sen, et al. "Learning deep structured semantic models for web search using clickthrough data." Proceedings of the 22nd ACM international conference on Information & Knowledge Management. ACM, 2013.

[12] Wang, Hao, Naiyan Wang, and Dit-Yan Yeung. "Collaborative deep learning for recommender systems." In Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1235-1244. ACM, 2015.

[13] He, Xiangnan, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. "Neural collaborative filtering." In Proceedings of the 26th International Conference on World Wide Web, pp. 173-182. International World Wide Web Conferences Steering Committee, 2017.

[14] Wei, Jian, Jianhua He, Kai Chen, Yi Zhou, and Zuoyin Tang. "Collaborative filtering and deep learning based recommendation system for cold start items." Expert Systems with Applications 69 (2017): 29-39.

[15] Sedhain, Suvash, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. "Autorec: Autoencoders meet collaborative filtering." In Proceedings of the 24th International Conference on World Wide Web, pp. 111-112. ACM, 2015.

[16] Li, Sheng, Jaya Kawale, and Yun Fu. "Deep collaborative filtering via marginalized denoising auto-encoder." In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 811-820. ACM, 2015.

[17] Strub, Florian, and Jeremie Mary. "Collaborative filtering with stacked denoising autoencoders and sparse inputs." In NIPS workshop on machine learning for eCommerce. 2015.

[18] Ouyang, Yuanxin, Wenqi Liu, Wenge Rong, and Zhang Xiong. "Autoencoder-based collaborative filtering." In International Conference on Neural Information Processing, pp. 284-291. Springer, Cham, 2014.

[19] Dong, Xin, Lei Yu, Zhonghuo Wu, Yuxia Sun, Lingfeng Yuan, and Fangxi Zhang. "A hybrid collaborative filtering model with deep structure for recommender systems." In Thirty-First AAAI Conference on Artificial Intelligence. 2017.

[20] Kramer, Mark A. "Nonlinear principal component analysis using autoassociative neural networks." AIChE journal 37, no. 2 (1991): 233-243.

[21] Vincent, Pascal, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. "Extracting and composing robust features with denoising autoencoders." In Proceedings of the 25th international conference on Machine learning, pp. 1096-1103. ACM, 2008.. .

[22] Rendle, Steffen, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. "BPR: Bayesian personalized ranking from implicit feedback." In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pp. 452-461. AUAI Press, 2009

[23] Yang, Shuang-Hong, Bo Long, Alexander J. Smola, Hongyuan Zha, and Zhaohui Zheng. "Collaborative competitive filtering: learning recommender using context of user choice." In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 295-304. ACM, 2011.

[24] DP Kingma, J Ba -Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014

[25] Wu, Yao, Christopher DuBois, Alice X. Zheng, and Martin Ester. "Collaborative denoising auto-encoders for top-n recommender systems." In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pp. 153-162. ACM, 2016.