中山大學 SUN YAT-SEN UNIVERSITY

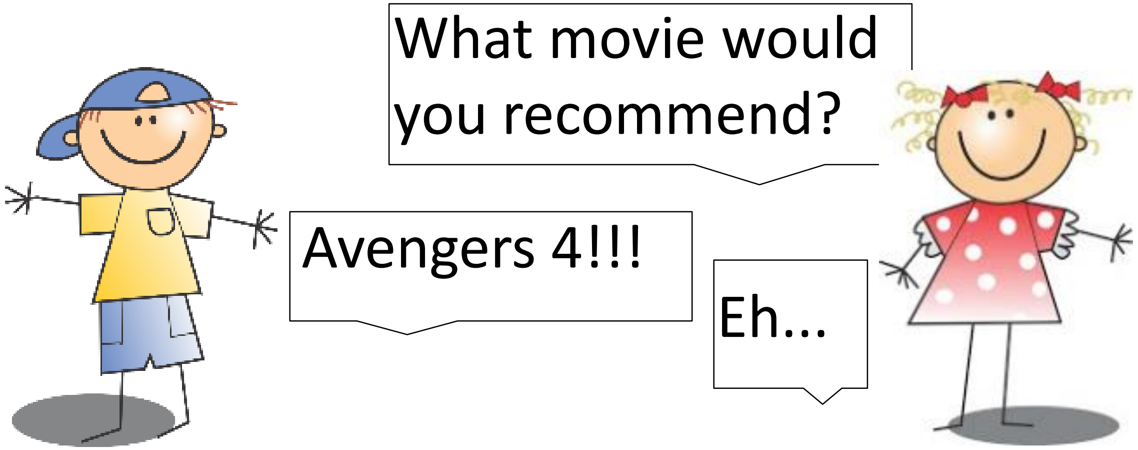
DeepCF: A Unified Framework of Representation Learning and Matching Function Learning in Recommender System

Zhi-Hong Deng, Ling Huang, Chang-Dong Wang, Jian-Huang Lai, and Philip S. Yu

AAAI 2019

Sun Yat-sen University

Introduction – Recommender Systems



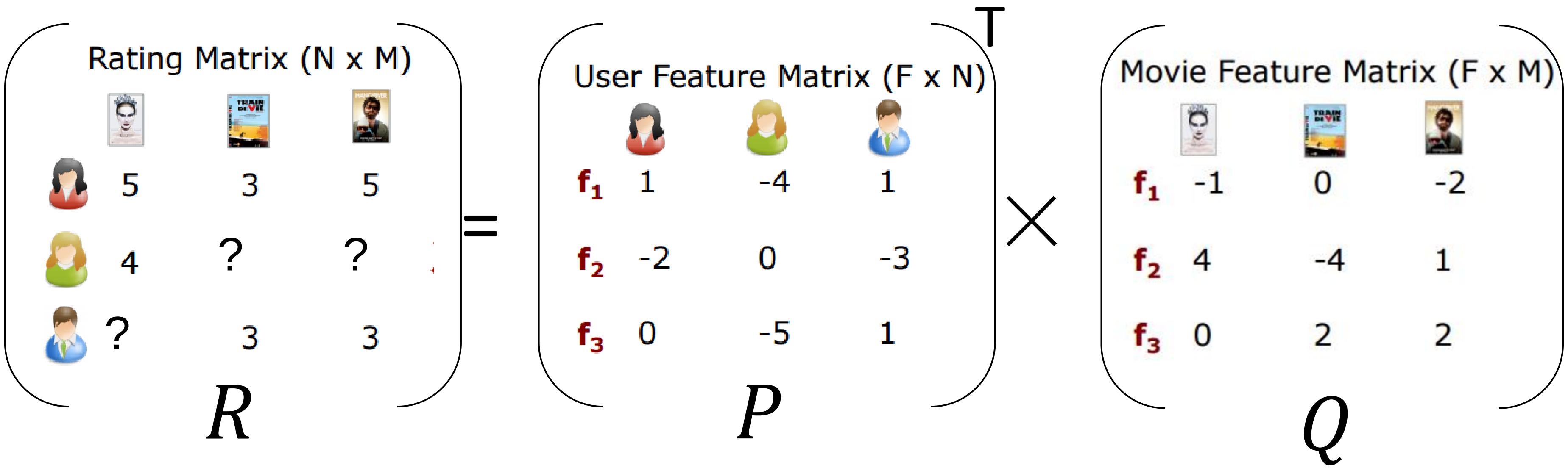
The old school way of recommendation.

Recommender Systems

The modern way of recommendation.

Introduction – Collaborative Filtering

- Collaborative Filtering is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating).
- Matrix factorization assumes users' preferences are controlled by some latent factors and the missing rating w.r.t. a user and an item can be inferred by computing their similarity in the latent space.



Predict: $\hat{r}_{ui} = p_u^T q_i$

Collaborative Filtering with Implicit Data

Rating Matrix R

5	3	5
4	?	?
?	3	3

→

Interaction Matrix Y

1	1	1
1	0	0
0	1	1

$$y_{ui} = \begin{cases} 1, & \text{if interaction (user } u\text{, item } i\text{) is observed;} \\ 0, & \text{otherwise.} \end{cases}$$

- We can assume y_{ui} obeys a Bernoulli distribution: $P(y_{ui} = k|p_{ui}) = \begin{cases} 1 - p_{ui}, & k = 0; \\ p_{ui}, & k = 1 \end{cases}$

$$= p_{ui}^k (1 - p_{ui})^{1-k},$$

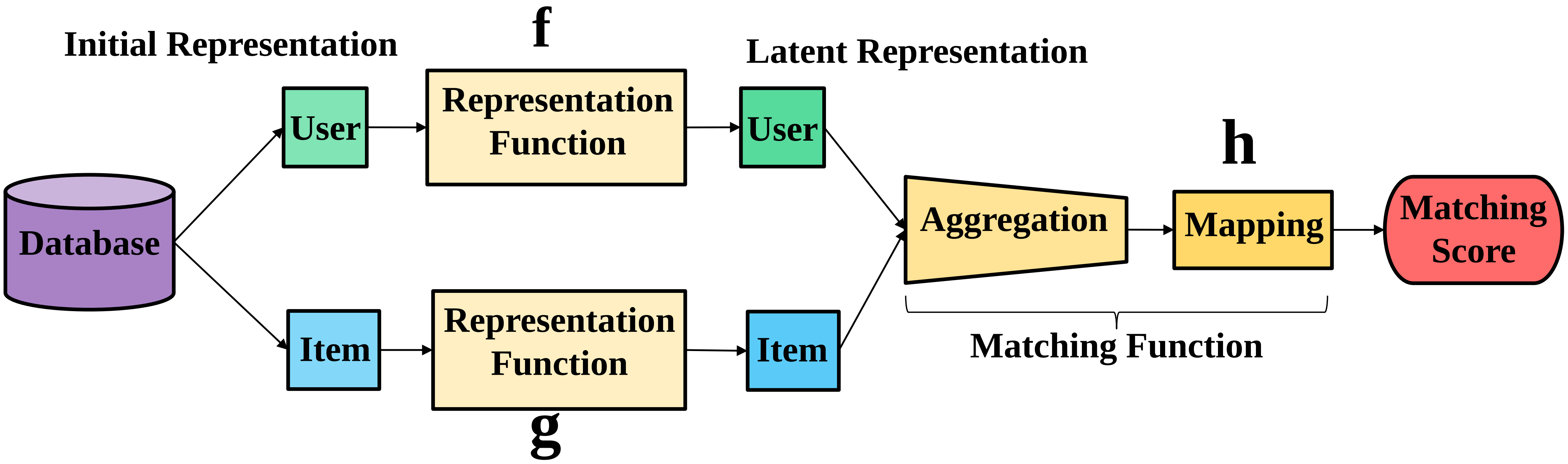
- We can define the likelihood function as: $L(\Theta) = \prod_{(u,i) \in \mathcal{Y}^+ \cup \mathcal{Y}^-} P(y_{ui}|\Theta)$

$$= \prod_{(u,i) \in \mathcal{Y}^+ \cup \mathcal{Y}^-} \hat{y}_{ui}^{y_{ui}} (1 - \hat{y}_{ui})^{1-y_{ui}},$$

- To maximized the likelihood is equivalent to minimize the binary cross-entropy.

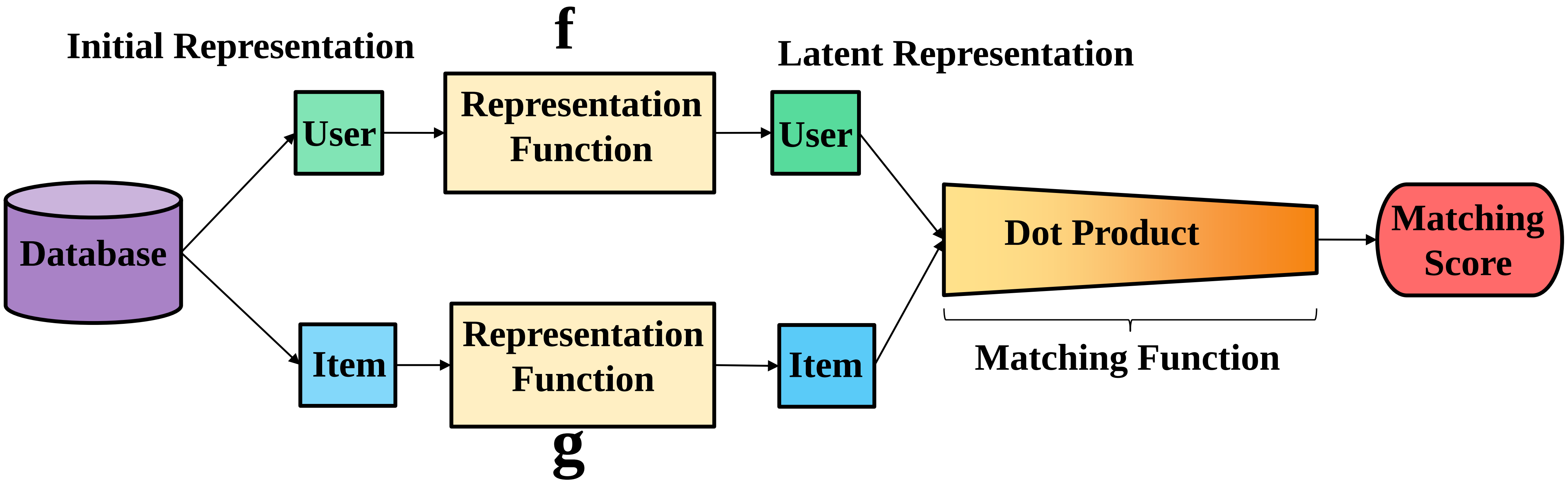
$$\ell_{BCE} = - \sum_{(u,i) \in \mathcal{Y}^+ \cup \mathcal{Y}^-} y_{ui} \log \hat{y}_{ui} + (1 - y_{ui}) \log(1 - \hat{y}_{ui})$$

The General Process of Collaborative Filtering



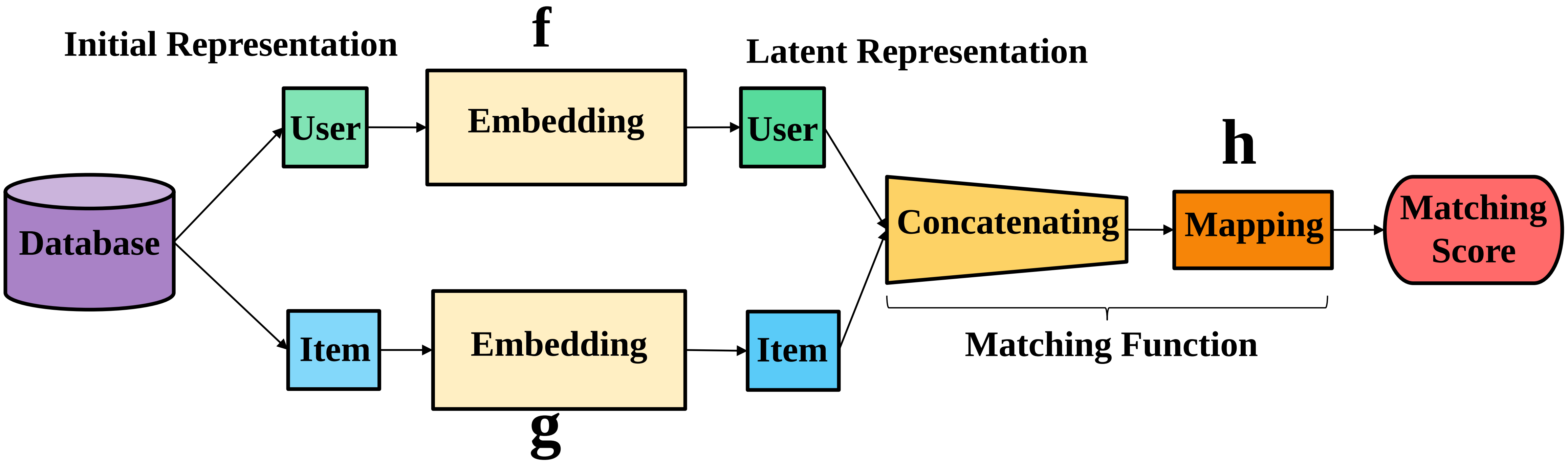
- Extracting data from the database.
- Calculate the latent representations for user and item.
- A non-parametric operation is performed to aggregate the latent representations.
- Use a mapping function h(·) to calculate the matching score.

The Representation Learning-based CF methods



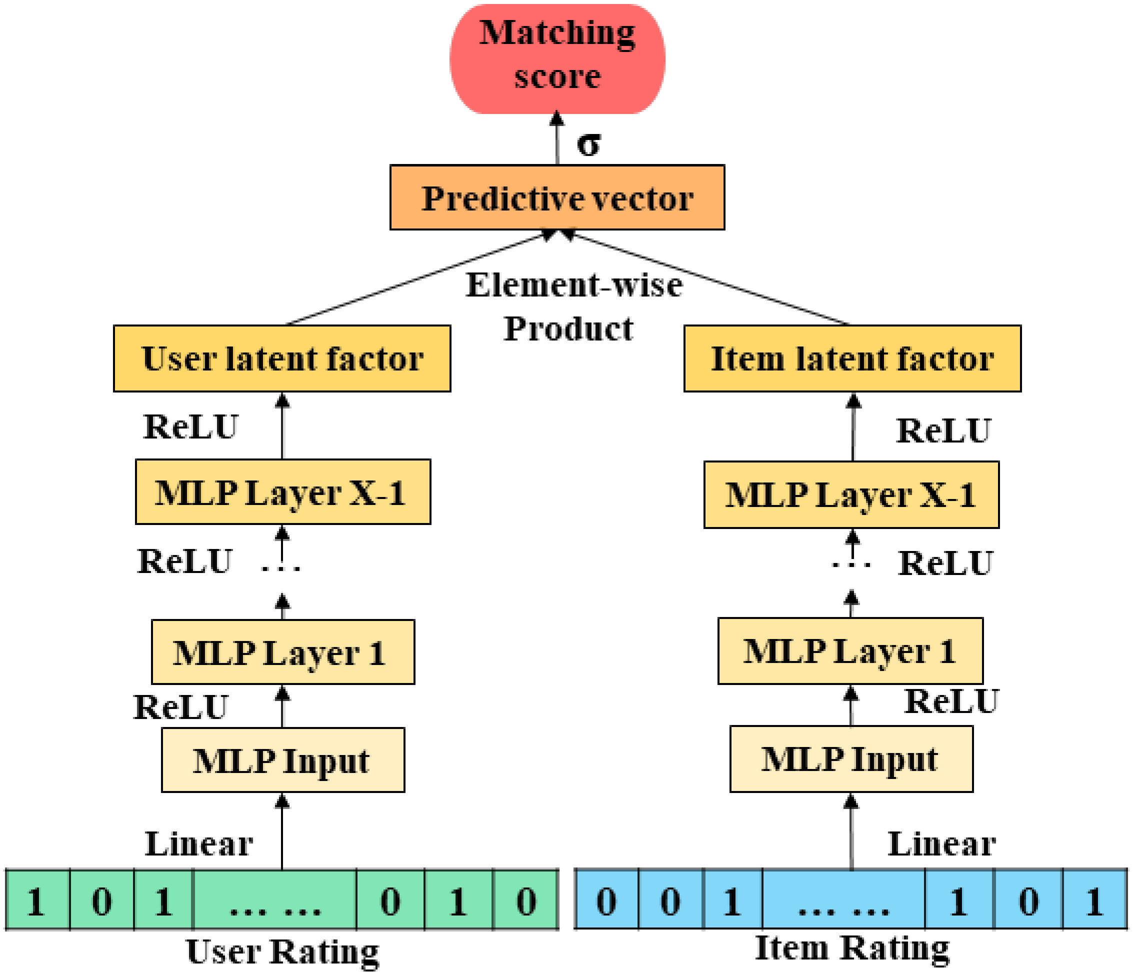
- Focusing on learning representation function while the matching function is usually assumed to be simple and non-parametric, i.e., dot product and cosine similarity.
- The model is supposed to learn to map users and items into a common space where they can be directly compared. This matches our prior knowledge.
- The latent factors are combined linearly which seriously limits the expressiveness.

The Matching Function Learning-based CF methods



- Focusing on learning matching function while the representation function is usually a simple linear embedding layer used to lower the dimensionality.
- Without additional assumption, using MLP to learn the matching function directly endows the model with a great flexibility.
- MLP is very inefficient in catching low-rank relations.

DeepCF: CFNet-rl Component



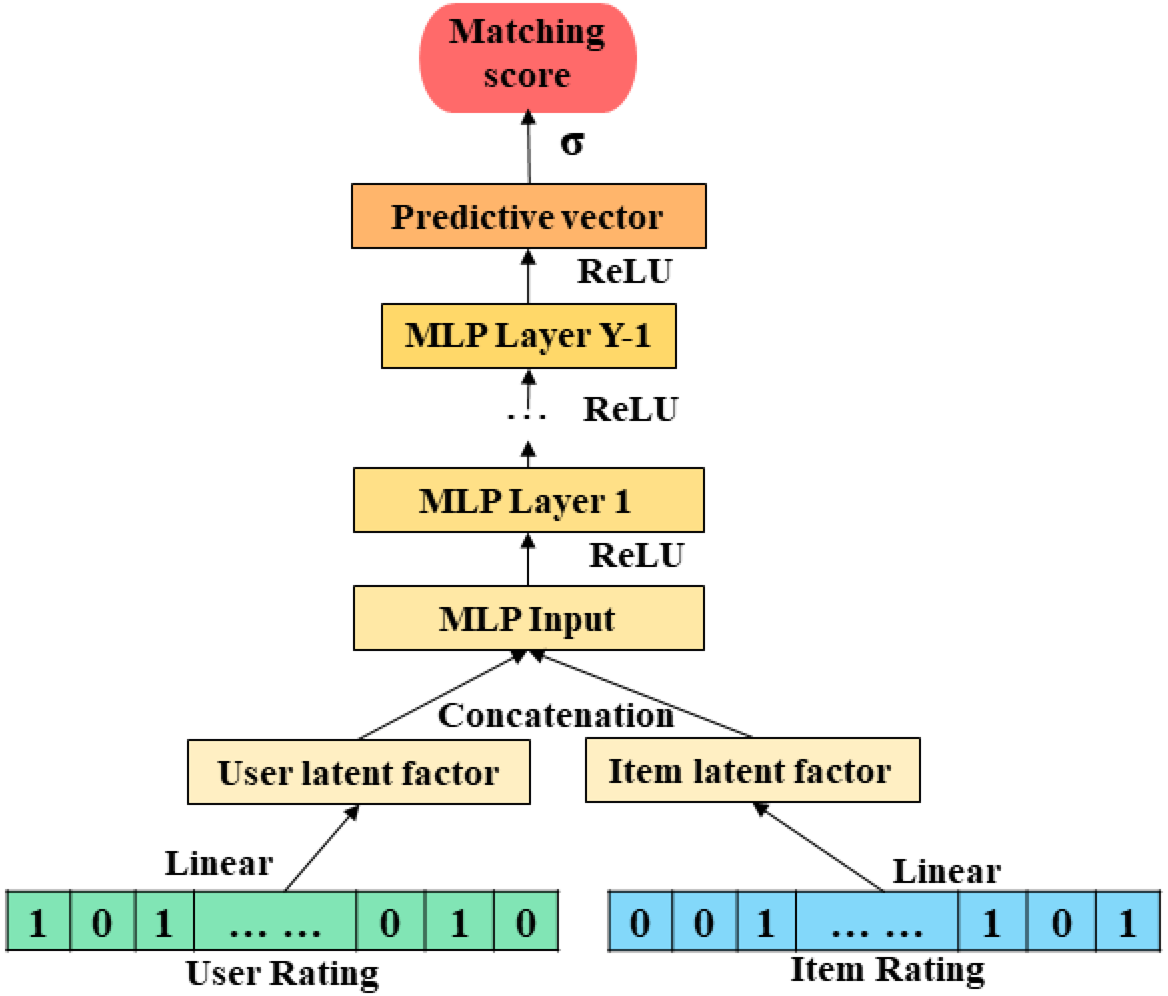
- The representation learning part for users can be defined as:

$$
\begin{aligned}
\mathbf{a}_0 &= \mathbf{W}_0^T \boldsymbol{y}_{u*} \\
\mathbf{a}_1 &= a(\mathbf{W}_1^T \mathbf{a}_0 + \mathbf{b}_1) \\
&\cdots\cdots \\
\mathbf{p}_u = \mathbf{a}_X &= a(\mathbf{W}_X^T \mathbf{a}_{X-1} + \mathbf{b}_X),
\end{aligned}
$$

- The latent representation for item is calculated in the same manner.
- Different from existed representation learning-based CF methods, the matching function part is defined as:

$$\hat{y}_{ui} = \sigma(\mathbf{W}_{out}^T(\mathbf{p}_u \odot \mathbf{q}_i)),$$

DeepCF: CFNet-ml Component



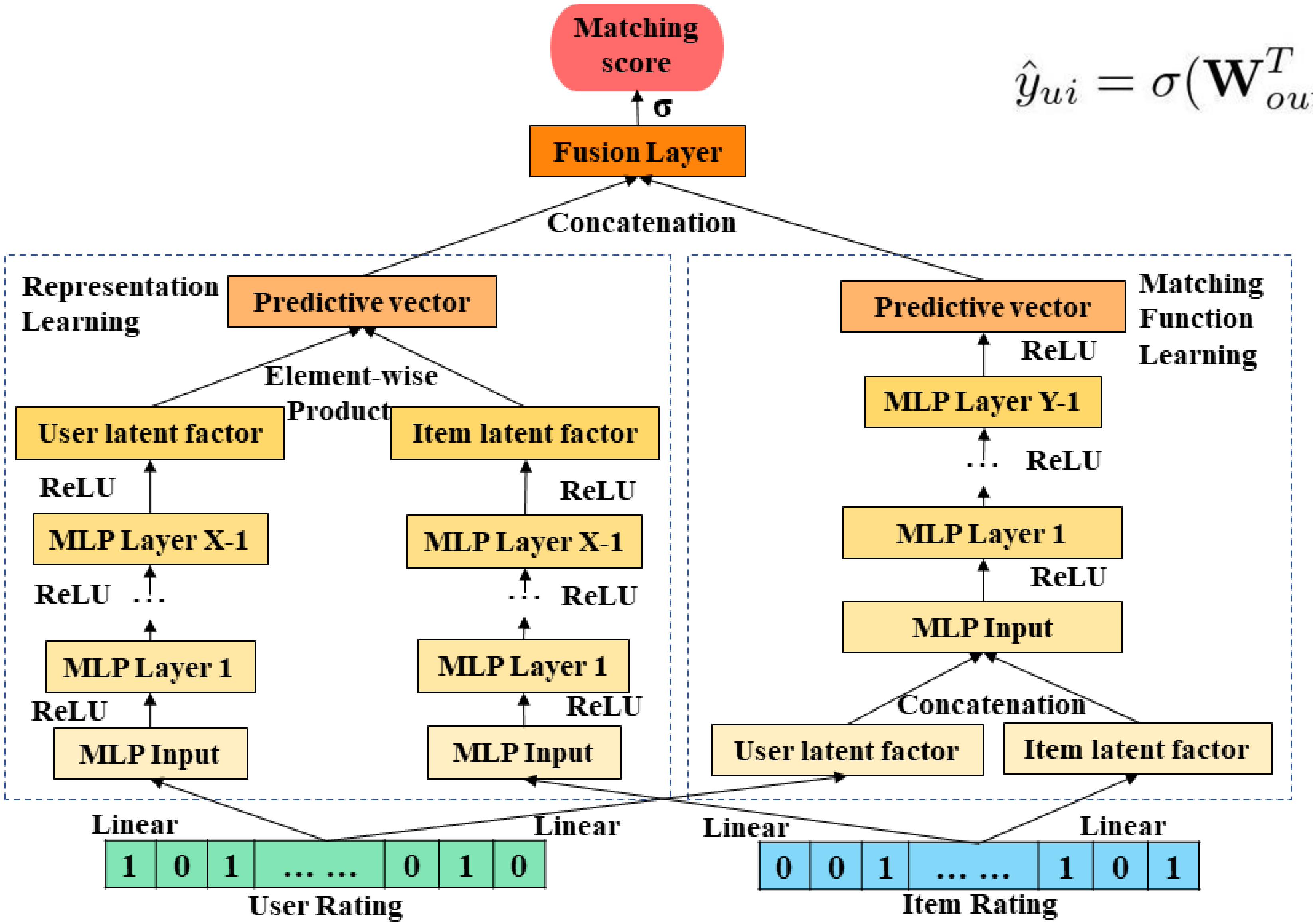
- The representation learning part is a simple linear embedding layer:

$$\mathbf{p}_u = \mathbf{P}^T \boldsymbol{y}_{u*}$$
$$\mathbf{q}_i = \mathbf{Q}^T \boldsymbol{y}_{*i}$$

- The latent representations and are aggregated by a simple concatenation operation and MLP is used as the mapping function:

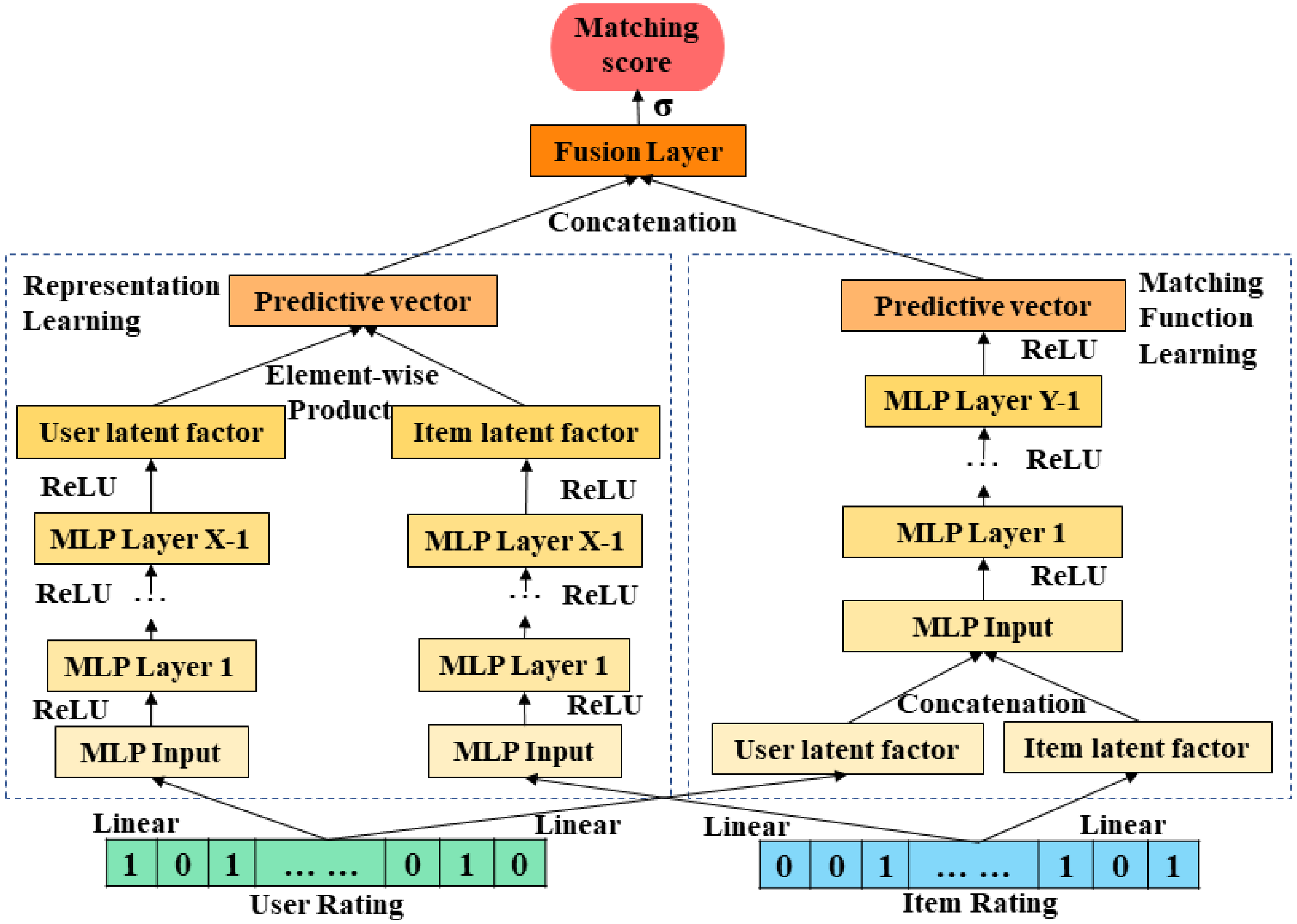
$$\mathbf{a}_0 = \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix}$$
$$\mathbf{a}_1 = a(\mathbf{W}_1^T \mathbf{a}_0 + \mathbf{b}_1)$$
$$\cdots\cdots$$
$$\mathbf{a}_Y = a(\mathbf{W}_Y^T \mathbf{a}_{Y-1} + \mathbf{b}_Y)$$
$$\hat{y}_{ui} = \sigma(\mathbf{W}_{out}^T \mathbf{a}_Y),$$

DeepCF: CFNet



$$\hat{y}_{ui} = \sigma(\mathbf{W}_{out}^T \begin{bmatrix} \mathbf{a}_Y^{rl} \\ \mathbf{a}_Y^{ml} \end{bmatrix}).$$

DeepCF: CFNet



- The objective function to minimize for the DeepCF framework is the binary cross-entropy function.
- To optimize the model, we use mini-batch Adam.
- Using pre-trained models to initialize the ensemble model can significantly increase the convergence speed and improve the final performance.

Experimental Settings

- Data Sets:

Statistics	ml-1m	lastfm	AMusic	AToy
# of Users	6040	1741	1776	3137
# of Items	3706	2665	12929	33953
# of Ratings	1000209	69149	46087	84642
Sparsity	0.9553	0.9851	0.9980	0.9992

- Evaluation Protocols：
  - We adopt the leave-one-out evaluation, i.e., the latest interaction of each user is used for testing.
  - Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) are used to evaluate the ranking performance.

Baselines

- **ItemPop** Items are ranked by their popularity, i.e., the number of interactions.
- **eALS** is a state-of-the-art MF method. It uses all unobserved interactions as negative instances and weights them non-uniformly by item popularity.
- **DMF** is a state-of-the-art representation learning-based MF method which performs deep matrix factorization with normalized cross entropy loss as loss function.
- **NeuMF** is a state-of-the-art matching function learning-based MF. It is the most related work to the proposed models. Different from our models, it adapts the deep+shallow pattern which has been widely adopted in many works such as (Cheng et al. 2016; Guo et al. 2017).

Experimental Results

Table 2: Comparison results of different methods in terms of NDCG@10 and HR@10.

Datasets	Measures	Existing methods				CFNet			Improvement of CFNet vs. NeuMF
		ItemPop	eALS	DMF	NeuMF	CFNet-rl	CFNet-ml	CFNet	
ml-1m	HR	0.4535	0.7018	0.6565	**0.7210**	0.7127	0.7073	**0.7253**	0.6%
	NDCG	0.2542	0.4280	0.3761	**0.4387**	0.4336	0.4264	**0.4416**	0.7%
lastfm	HR	0.6628	0.8265	0.8840	**0.8868**	0.8840	0.8834	**0.8995**	1.4%
	NDCG	0.3862	0.5162	0.5804	**0.6007**	0.6001	0.5919	**0.6186**	3.0%
AMusic	HR	0.2483	0.3711	0.3744	0.3891	0.3947	**0.4071**	**0.4116**	5.8%
	NDCG	0.1304	0.2352	0.2149	0.2391	**0.2504**	0.2420	**0.2601**	8.8%
AToy	HR	0.2840	0.3717	0.3535	0.3650	0.3746	**0.3931**	**0.4150**	13.7%
	NDCG	0.1518	**0.2434**	0.2016	0.2155	0.2271	0.2293	**0.2513**	16.6%

- CFNet achieves the best performance in general and obtains high improvements over the state-of-the-art methods. Most importantly, such improvement increases along with the increasing of data sparsity, where the datasets are arranged in the order of increasing data sparsity.

- Replacing the non-parametric cosine similarity with element-wise product and a parametric layer significantly improves the performance.

Without pre-training v.s. With pre-training

Table 3: Comparing the performance of CFNet with and without pre-training.

Datasets	Without pre-training HR	Without pre-training NDCG	With pre-training HR	With pre-training NDCG
ml-1m	0.6962	0.4222	0.7253	0.4416
lastfm	0.8685	0.5920	0.8995	0.6186
AMusic	0.3530	0.2204	0.4116	0.2601
AToy	0.3067	0.1653	0.4150	0.2513

- The pre-training process which ensures CFNet-rl and CFNet-ml to learn features from different perspectives and therefore allows the model to generate better results.

Sensitivity Analysis of Hyperparameters

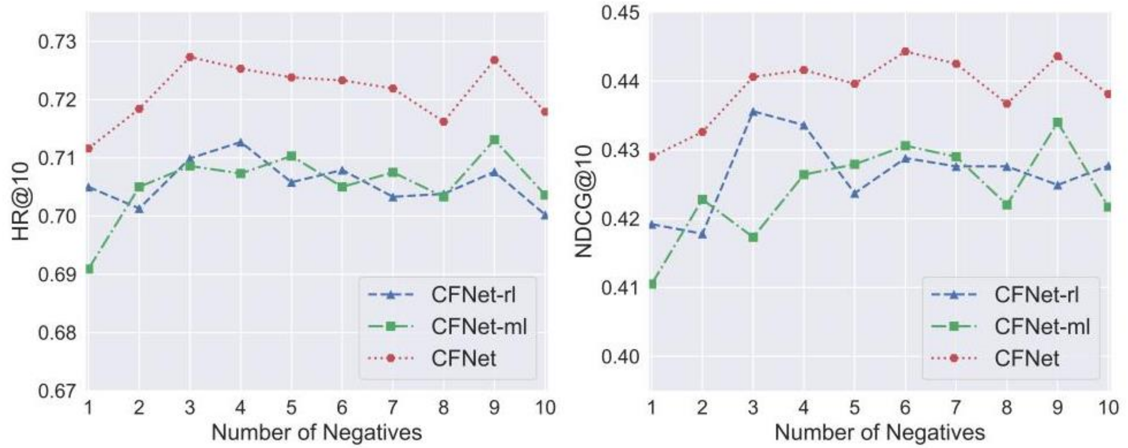


Figure 3: The effect of negative sampling ratio on performance on the ml-1m dataset.

Table 4: Comparing the performance of CFNet with different number of predictive factors.

Datasets	Measures	Dimensions of predictive vectors 8	16	32	64
ml-1m	HR	0.6820	0.6982	0.7157	0.7253
	NDCG	0.3992	0.4161	0.4351	0.4416
lastfm	HR	0.8840	0.8857	0.8937	0.8995
	NDCG	0.6049	0.6111	0.6143	0.6186
AMusic	HR	0.4003	0.4313	0.4262	0.4116
	NDCG	0.2480	0.2617	0.2661	0.2601
AToy	HR	0.3797	0.3902	0.4026	0.4150
	NDCG	0.2273	0.2331	0.2383	0.2513

- Sampling merely one or two instances is not enough.

  Sampling too many negative instances is harmful.

  Overall, the optimal sampling ratio is around 3 to 7.

- More predictive factors usually lead to better performances since it endows the model with larger capability and greater ability of representation

Conclusion

- We point out the significance of incorporating collaborative filtering methods based on representation learning and matching function learning, and present a general Deep Collaborative Filtering (DeepCF) framework. The proposed framework abandons the traditional Deep+Shallow pattern and adopts deep models only to implement collaborative filtering with implicit feedback.

- We propose a novel model named Collaborative Filtering Network (CFNet) based on the vanilla MLP model under the DeepCF framework, which has great flexibility to learn the complex matching function while being efficient to learn low-rank relations between users and items.

- We conduct extensive experiments on four real-world datasets to demonstrate the effectiveness and rationality of the proposed DeepCF framework.

Future Work

- Keep improving and extending our code.[1]

- Auxiliary data can be used to further improve the initial representations of users and items. Richer information usually leads to better performance.

- Exploring a better way to incorporate the two types of CF methods.

- Although we use DeepCF to solve the top-N recommendation problem with implicit data, it's also suitable for other data mining tasks that try to match two kinds of entities.

1. https://github.com/familyld/DeepCF

Thank you!