Critically Examining the Claimed Value of Convolutions over User-Item Embedding Maps for Recommender Systems

Maurizio Ferrari Dacrema
Politecnico di Milano, Italy
maurizio.ferrari@polimi.it

Federico Parroni
Politecnico di Milano, Italy
federico.parroni@mail.polimi.it

Paolo Cremonesi
Politecnico di Milano, Italy
paolo.cremonesi@polimi.it

Dietmar Jannach
University of Klagenfurt, Austria
dietmar.jannach@aau.at

ABSTRACT

In recent years, algorithm research in the area of recommender systems has shifted from matrix factorization techniques and their latent factor models to neural approaches. However, given the proven power of latent factor models, some newer neural approaches incorporate them within more complex network architectures. One specific idea, recently put forward by several researchers, is to consider potential correlations between the latent factors, i.e., embeddings, by applying convolutions over the user-item interaction map. However, contrary to what is claimed in these articles, such interaction maps do not share the properties of images where Convolutional Neural Networks (CNNs) are particularly useful. In this work, we show through analytical considerations and empirical evaluations that the claimed gains reported in the literature cannot be attributed to the ability of CNNs to model embedding correlations, as argued in the original papers. Moreover, additional performance evaluations show that all of the examined recent CNN-based models are outperformed by existing non-neural machine learning techniques or traditional nearest-neighbor approaches. On a more general level, our work points to major methodological issues in recommender systems research.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → **Neural networks**.

KEYWORDS

Deep Learning; Evaluation; Convolutional Neural Networks

ACM Reference Format:
Maurizio Ferrari Dacrema, Federico Parroni, Paolo Cremonesi, and Dietmar Jannach. 2020. Critically Examining the Claimed Value of Convolutions over User-Item Embedding Maps for Recommender Systems. In *The 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3340531.3411901

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3411901

1 INTRODUCTION

In the era of exponential information growth, recommender systems have proven to be valuable tools to help users explore the vast number of options at their disposal. Over the last two decades, a large number of collaborative filtering algorithms was proposed for item ranking and relevance prediction, from early nearest-neighbors approaches to machine learning models [16, 32, 34, 37, 39]. Among these techniques, *matrix-factorization* (MF) algorithms were particularly popular in the past decade after the Netflix Prize competition [4], both in industry and academia. These algorithms project users and items into a low dimensional latent space [18, 19], and an interaction between a user and an item is usually modeled as the dot product of their respective latent vectors.

In recent years, the attention of researchers and industry has moved to deep learning (neural) approaches for collaborative filtering, and various network architectures have been proposed, e.g., [6, 14, 49]. In several of these approaches, researchers try to incorporate specific architectural components into their models that were previously found to be effective in other application domains of deep learning. Examples of such architectural components are attention layers, autoencoders and convolution layers [24, 49]. In particular Convolutional Neural Networks (CNNs) were successfully applied to different recommendation-related tasks, including image or text feature extraction for content-based models [42], sequential recommendations [40] or collaborative filtering [49].

In some recent works, different proposals were also put forward to combine the proven power of low dimensional approximation models (e.g., latent factor models) with CNNs. In particular, one underlying idea of the proposals in [9, 13, 43, 48] is to use CNNs to model and leverage correlations in the latent factors (embeddings) space. In all these papers, the respective authors claim to have obtained significant gains in accuracy by applying convolutions over user-item interaction maps derived from the outer product of user-item embeddings.

In some of these works—all of them originally published at the highly-ranked IJCAI conference—the authors argue that these interaction maps can be viewed as analogous to images, an application area in which CNNs are very effective. However, user-item interaction maps, as produced by common latent space approaches, do not share the properties of images. For most common latent space models, there is no natural order of the elements in the latent vectors, the dimensions are independent, and the correlation between the latent dimensions is actually not modeled. Therefore, it is more

than surprising that the above-mentioned CNNs approaches were able to benefit from detecting correlations between the dimensions.

In this work, we therefore critically examine the progress claimed in these papers, based on both theoretical considerations and experimental evaluations. In particular, we report the results of ablation studies that were not present in the original papers. These results indicate that the proposed CNN-based models do *not* capture correlations between the embedding dimensions as claimed in the original papers. Rather, they merely act as a non-linear function of their element-wise product and the removal of the embedding correlations leads to no significant effects. Our work therefore points to major issues in the way these articles have justified and demonstrated their claims. A particular problem in that context may lie in the *missing validation of the assumptions* these models rely upon.[1]

The question however still remains how the authors were able to demonstrate significant performance gains in their experimental evaluation. One reason might lie in the choice of the baselines that were used in their evaluation. If the baselines were not strong, demonstrating a "win" over previous methods might be possible even if the added CNN layer merely acts as an additional universal approximator function. We have therefore conducted additional experiments, using the exact same experimental setup as in the original papers, where we benchmarked the new methods against established non-neural machine learning models and traditional nearest-neighbor techniques. Our results show that for all algorithms, datasets, and metrics, existing techniques outperformed the recent CNN-based methods. We share the code and data used in our experiments online.[2]

On a more general level, our work adds to the growing evidence that we, as a community, face significant methodological problems, which makes it difficult to judge if we are truly moving the field much forward. Previous works in the area of recommender systems [11, 28, 36], information retrieval [2, 44] or time-series forecasting [30] found that the progress achieved with certain complex models is sometimes non-existent and was only observed because the chosen baselines were weak or not properly optimized. In another recent paper Rendle et al. [35] confirmed previous results that NCF [14] is not able to consistently outperform non-neural baselines [10, 11], and showed it is not trivial to learn a dot product with a multilayer perceptron. These works indicate that sometimes the experimental evaluations are not well suited to demonstrate the authors' claims regarding which part of a complex architecture actually contributes to an observed performance gain. The same observation was recently made by Lipton et al. [26], where it was argued that sometimes papers present complex algorithms involving several components (e.g., architecture, preprocessing steps, training procedure) without reporting any ablation study to clarify the contribution of the individual components. The article encouraged authors to ask "What worked?" and "Why?" rather than just "How well?", highlighting the importance of sound empirical inquiry, which can yield new knowledge or insights even when no new algorithm is proposed.

[1] A clear explanation of such underlying assumptions is also stipulated in the *Machine Learning Reproducibility Checklist* (https://www.cs.mcgill.ca/~jpineau/ReproducibilityChecklist-v2.0.pdf), which was, for example, adopted in recent years as part of the NeurIPS submission process.

[2] https://github.com/MaurizioFD/RecSys2019_DeepLearning_Evaluation

With our study, we address such questions and encounter signs that the lack of theoretical underpinnings of new deep learning approaches may contribute to the observed problems.

2 BACKGROUND

2.1 Principles and Assumptions of CNNs

A convolutional neural network is a multilayer feed-forward neural network that was originally developed to address image recognition problems [47]. Unlike other types of neural networks, CNNs were therefore designed for certain types of inputs, e.g., images, which have a specific topology. As stated in [20], the paper describing the seminal *AlexNet* model for image classification, CNNs are based on two strong and important assumptions regarding the nature of the processed input data (i.e., the images): *stationarity* of statistics and *locality* of pixel dependencies, that is, pixels that are close will be strongly correlated [21].

With data exhibiting these properties, local features (e.g., lines, corners) emerge from their respective immediate surroundings, regardless of their absolute position in the observed data. CNNs have also been proven effective in several other scenarios where the data exhibits feature locality (e.g., time series forecasting, natural language processing). The importance of translation invariance and feature locality for CNNs has been widely discussed, both in terms of *spatial locality* [22, 27, 41, 45], and *time locality* for sequence modeling [3].

Technically, CNNs in their traditional form consist of a convolution layer and a pooling layer.[3] The convolution layer uses a kernel, with certain weights, which is moved across the two dimensional feature matrix (i.e., the image). Sharing the kernel weights across the image allows a CNN to have much fewer parameters than a fully connected NN and to leverage the spatial locality and location invariance of features. The pooling layer reduces the dimensionality of the data, allowing successive convolution layers on a broader field of view. Multiple convolutional layers are then able to interpret increasingly complex patterns by further aggregating lower-level features. As stated in [21], after the detection of a feature, only the relative position of that feature with respect to other features is relevant, not the absolute one.

Convolution is usually applied on a *local area* (i.e., the kernel size) of a two-dimensional map of a certain size. Identifying the *local area* on which to apply the convolution requires a definition of *proximity* between points. Depending on the use case, different definitions of proximity may be used. In case of images, the proximity is defined in spatial terms, i.e., pixels that are close in the image will be perceived as close by an observer and are therefore meaningful for the reconstruction of more complex patterns. Other definitions of proximity have also been developed for non-Euclidean data like social networks or knowledge graphs [8].

2.2 Use of CNNs in Recommender Systems

A growing number of papers aim to use CNNs for collaborative filtering tasks. Most existing approaches can be grouped into three categories [49]:

[3] The use of max-pooling to reduce the dimensionality of the data has been recently criticized, see [38].

Feature extraction: In this case, a CNN is used to extract features from heterogeneous data sources, e.g., images, video, audio, which are then used in another recommendation model [42].

Pretrained embeddings: In such approaches, a CNN is applied on user or item embeddings that were pretrained by another model, e.g., [13, 43].

Learnable embeddings: Also in this case the CNN is applied on user or item embeddings. Differently from the previous case, the embeddings are an integral part of the model and are trained along with the CNN (i.e., they are not pre-trained with another approach), e.g., in [17, 48].

In this paper we will focus on the last two cases, when CNNs are applied on embeddings.[4]

If we compare images or graph data to embeddings in the specific form of latent factors derived from a user-item rating matrix, there is a key difference. For images or graphs, there is a "natural" way of defining the local area, e.g., based on the distance in pixels or the number of hops in the graph. The corresponding proximity measure has a strong relation with the data semantics.

However, the papers analyzed in this work, i.e., [9, 13, 43, 48], do not clearly provide a definition of locality for the embeddings, nor do they describe the semantic topology of the input data. In the ConvNCF approach [13], for example, the input to the CNN is a user-item interaction map that is created by computing the outer product of embeddings pretrained using matrix factorization. While the authors argue that this map is analogous to an image and therefore the use of CNN is justified, they do not demonstrate or discuss in detail what the resulting map topology should represent and whether it possesses typical image properties (e.g., spatial locality and translation invariance). Technically, the interaction map created by the outer product of the embeddings in the described approach contains two components: (i) the main diagonal that represents the element-wise product between two embedding vectors and (ii) the off-diagonal elements that capture embeddings correlations. Unfortunately, none of the analyzed papers measured and compared the contribution of the two components to the proposed CNN model. In all papers the authors claim that the CNN is able to model the correlations between embeddings based on the fact that CNN models outperform other models that are not using convolutions. This, we argue, is not sufficient. Comparing models with vastly different structures means that a multitude of factors may have an impact on the results. Ultimately, it is not entirely clear from the provided experiments how much the correlations between embeddings, as modeled by the CNN, actually contributed to the observed performance gains.

3 OVERVIEW OF ANALYZED APPROACHES

In this paper we examine three recent neural approaches that use convolutions over embeddings derived from a user-item rating matrix. All approaches were published at previous IJCAI conferences and for all of them the source code was available. We identified an additional relevant work [17]. Its experimental setting (i.e., source code and data) was however not reproducible based on the material provided by the authors.

3.1 Convolutional Neural Collaborative Filtering

Convolutional Neural Collaborative Filtering (*ConvNCF*) was proposed in [13]. The ConvNCF model is trained in two steps. First, a matrix-factorization model is fitted on the data. Then, for each user-item interaction, the outer product of their embedding is computed, resulting in a two-dimensional *interaction map* on which the CNN is applied.

Following the original notation, let u be a user and i an item, $P \in \mathbb{R}^{M \times K}$ and $Q \in \mathbb{R}^{N \times K}$ the embedding matrix of users and items, respectively; K the embedding size, M the number of users and N the number of items. Lastly, let $p_u, q_i \in \mathbb{R}^K$ be their respective embeddings. Based on these embeddings, the interaction map $E \in \mathbb{R}^{K \times K}$ is obtained by computing their outer product as follows:

$$E = p_u \oplus q_i = p_u q_i^T$$
$$e_{x,y} = p_{u,x} \cdot q_{i,y} \tag{1}$$

In the original paper, pretraining is performed with an MF BPR model [34] and, in a subsequent paper, the authors extend the pretraining to FISM and SVD++ models [9]. As mentioned before, the interaction map is said to be analogous to an image, but there is no deeper discussion of this claim, which is therefore not verified, and the CNN is said to model embedding correlations but no direct measurement of their contribution is provided.

3.2 Convolutional Factorization Machines

Convolutional Factorization Machines (*CFM*) were proposed in [43] as a context-aware model able to overcome the limited modeling capacity of Factorization Machines (FM) [33], which are constrained by a linear representation of feature interactions. Similarly to ConvNCF, CFM applies a convolution operation on the outer product of the embeddings. In the CFM approach, however, the embeddings are obtained from a FM via a self-attention layer, which reduces the model's dimensionality. The outer product of the embeddings is computed independently for each context and all interaction maps are stacked on top of each other, forming an interaction cube, on which 3D convolution is applied. If C is the number of contextual features, the interaction maps in the memory cube will be $C(C-1)/2$.

Although the scenario of application is different from ConvNCF and the embeddings are obtained from a different pretraining step, CFM has the same theoretical problems as ConvNCF in that the outer product is treated as if it were an image, without ever demonstrating that this assumption holds. The ability of CFM to model embeddings interaction is again claimed based on it outperforming baselines with quite different architectures, including ConvNCF.

3.3 Coupled Collaborative Filtering

Coupled Collaborative Filtering (*CoupledCF*) was proposed in [48] as a method to learn implicit and explicit couplings between users and items, taking advantage of them not being independent, to leverage side information more effectively (e.g., user demographics,

[4] Note that we only consider approaches that use a user-item rating matrix as an input. CNNs were also applied for session-based recommendation [46], where they however showed some limitations as well [29].

item features). The model is composed of two cooperating architectures, one is a deep collaborative filtering model which only uses user-item interactions, while the other is a CNN on user and item embeddings whose aim is to learn the couplings.

CoupledCF is different from the other two methods examined here in that the embeddings are not pretrained but are parameters of the model to be learned. In [48], the authors show experimentally that the quality of the model is improved by adding the CNN and claim this demonstrates that the model is effectively learning the couplings. Again, as previously mentioned, the paper does not distinguish between the contribution of the the embeddings correlation and the element-wise product.[5]

4 ANALYSIS

One fundamental assumption of the analyzed papers is that the interaction map computed via an outer product is analogous to an image (i.e., exhibits spatial locality and translation invariance). In this section, we demonstrate why this is not the case. We will first discuss this aspect theoretically and then present the results of two empirical studies. In the first study we show that changing the input topology (i.e., the ordering of the latent factors) does not have an impact on accuracy. Clearly, if the interaction map had local features a significant drop in accuracy would be expected. The second study consists of two ablation analyses showing that the correlations between embeddings do not provide a statistically significant contribution to the accuracy of the CNN models.

4.1 Theoretical Considerations

As previously stated, CNNs were developed to model data that exhibits feature locality and a strong topology. To assess whether CNNs are an appropriate tool to be used on interaction maps, we first have to analyze what constitutes the topology in an interaction map and why two points are within or outside each others' local area.

Consider three cells of an interaction map E, created via Equation 1, with coordinates (x, y), $(x, y+1)$, $(x, y+4)$. If we consider point (x, y), with a kernel size of 2, $(x, y+1)$ will be in its local area while $(x, y+4)$ will not. But what is the relation between y, $y+1$ and $y+4$? If the embeddings are created with a typical latent factor model (e.g., MF or FM), as done in two of the analyzed papers, the answer is that the latent factors y and $y+1$ are direct neighbors in the embedding vector, while y and $y+4$ are not.

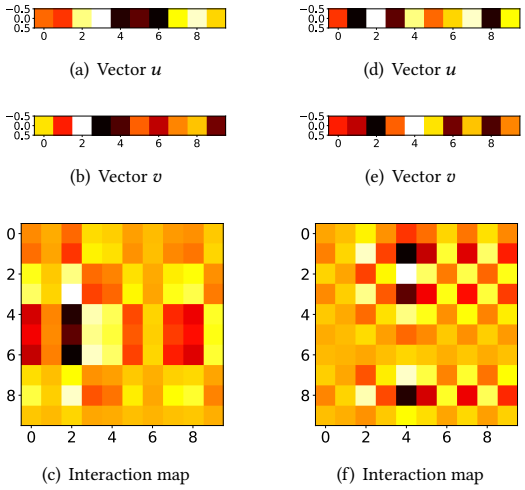
The question now is whether the position of the latent factors has a specific meaning. If we look at typical matrix-factorization algorithms, such as MF BPR or iALS [16], we can see that a prediction is computed as follows [34].

$$\hat{r}_{ui} = p_u^T \cdot q_i = \sum_k^K p_{u,k} \cdot q_{i,k} \tag{2}$$

Here, $\hat{r}_{ui}$ is the predicted relevance for user u on item i, p_u and q_i are the user and item embedding vectors, and k is the latent factor index.

From Equation 2, we see that the ordering of the latent factors has no impact on the prediction, and the model only requires a biunivocal correspondence between the columns of P and Q, regardless of their relative ordering. Such a lack of a *natural ordering* of the latent factors is common to many matrix-factorization algorithms including MF BPR, AsySVD, and iALS [16, 34]. Only for some techniques, like PureSVD [7], the latent factors are ordered according to the decreasing singular value they are associated with. Due to this lack of a natural ordering, the specific arrangement of the latent factors is mainly a contingent property and a multitude of equivalent models can be learnt from the same data due to the stochastic nature of the training process. Each permutation of the ordering of the factors leads to an equivalent MF model but to a different interaction map, which will exhibit different local features.

The effects of permutating the columns of the vectors can also be analyzed visually. Consider two randomly created vectors u and v of length 10 (Figures 1.a and 1.b) and two permutations of the same vectors (Figures 1.d and 1.e), where darker cells indicate higher values. It can be easily seen that the interaction map of the original vectors (Figures 1.c) and the one of the permuted vectors (Figure 1.f) do not have any identifiable pattern in common.



(a) Vector u (d) Vector u

(b) Vector v (e) Vector v

(c) Interaction map (f) Interaction map

Figure 1: Effects of permutating the columns of vectors u and v on their resulting outer product (the interaction map).

This lack of a natural ordering provides evidence that the interaction map is not analogous to an image because it does not exhibit spacial locality (i.e., meaningful local features).

4.2 Experiment Configurations

We conducted three types of computational experiments.[6] Two of them, discussed in Section 4.3 and Section 4.4, were designed to provide evidence that convolutions over the user-item interaction map do not lead to the claimed effects. The third, discussed in Section 4.5, shows that the analyzed CNN-based methods are consistently outperformed by existing non-neural techniques, which points to a problem in how the baselines were selected and optimized in the original works.

[5] An additional issue regarding the choice and optimization of the baselines used in [48] was recently reported in [10].

[6] We share the code and data used in our experiments online: https://github.com/MaurizioFD/RecSys2019_DeepLearning_Evaluation

In all our studies, we used the same experimental designs as in the original papers. In particular, we used the original code, data, data splits, as well as hyperparameters that were provided by the authors.[7] To determine the number of epochs, which is not usually reported, we apply early-stopping, see also [11]. The training is stopped if the recommendation quality does not improve for 5 consecutive evaluation steps. The evaluation setup for each method was as follows.

ConvNCF was evaluated on a dataset from Yelp. The algorithm code and the data split for the Yelp data was published by the authors, but not the preprocessing code. We preprocessed the data based on the information in the paper. MF BPR was used to pretrain the latent factors.

CFM was tested on a music dataset from Last.fm. The code and the preprocessed data split for Last.fm were provided by the authors. The latent factors were pretrained using a Factorization Machine.

CoupledCF was tested on the Movielens1M dataset, which also contains side information about users and items. CoupledCF, as stated above, does not use pretrained models but learnable embeddings.

Note that in the original papers additional evaluations with similar outcomes were reported for alternative datasets. For the purpose of this present study, which aims to show that CNNs in principle cannot work as claimed, it is sufficient to provide one counterexample. Therefore, we limit our discussions to one of the datasets that was used in the original paper.

In all original papers the authors use a leave-one-out evaluation procedure. In two cases a number of randomly sampled negative items (e.g., 99 for CoupledCF) were ranked with the true positive. The Hit Rate (HR) and the NDCG are used as evaluation measures in all papers, using different cut-off lengths.[8] In our evaluation we applied, for each algorithm, the exact same evaluation setting as described in the original paper.

4.3 Varying the Input Topology

In our first experiment, we varied the topology of the inputs (i.e., the order of the latent factors) which are fed to the CNN. Given our theoretical considerations from Section 4.1, altering the topology should have virtually no impact on the model quality because the topology cannot provide any relevant information. To empirically validate this consideration, we designed the following experiment for the approaches that use pretrained embeddings (ConvNCF and CFM).

First, we pre-train the embeddings as done in the original articles, i.e., using either MF BPR or FM. We then create 20 equivalent pretrained models by randomly permutating the positions of the latent factors. Each permutation is applied consistently on the user and item latent factors in a way that the latent factor model remains equivalent. Each permutation will lead to different interaction maps. For each of these permutations, a convolution model is trained, and the quality of each resulting model is evaluated based on the measures used in the original papers (HR and NDCG) at cutoff 10.

	NDCG	HR
MF BPR	0.1576 ± 0.0000	0.2966 ± 0.0000
ConvNCF	0.1623 ± 0.0008	0.3052 ± 0.0019
FM	0.1230 ± 0.0000	0.2234 ± 0.0000
CFM	0.1730 ± 0.0398	0.3155 ± 0.0724

Table 1: Averaged performance results and standard deviations obtained for 20 permutations of the interaction maps.

The results of this experiments are reported in Table 1. We both report *(i)* the results for the model without the CNN layer (MF BPR and FM) and *(ii)* the results for the full model (ConvNCF and CFM). For each algorithm, we report the averaged accuracy and the standard deviation resulting from the 20 permutations. The following observations can be made.

- For the "plain" MF BPR and FM models, the standard deviation is zero, as expected by definition from Equation (2). However, also for the CNN-based models, the standard deviation is almost zero. This confirms that changing the input topology has no relevant impact on the results, indicating that the order of the latent factors, as expected, does not matter and no local features can exist in the interaction map. Note that statistical tests like the t-test cannot be applied when the variance is zero.
- The CNN models show improved accuracy over the pretraining models. For the ConvNCF method, the gains are tiny, i.e., we could not reproduce in our experiments that the CNN adds much value.[9] For the CFM model, which applies a more complex preprocessing step, improvements over the plain FM model *could* be reproduced. These gains, however, cannot be attributed to the fact that the interaction maps can be considered as images (given the irrelevance of the ordering of the "pixels").

To put these observed gains into perspective, we, as mentioned above, executed additional experiments in which we compared the performance of CFM and other methods with existing non-neural techniques, see Section 4.5, The results are in line with previous observations in [11] and show that the gains happen at a performance level that is largely below the performance of a traditional ItemKNN method [39] (see Table 5).

4.4 Ablation Studies

Despite being on a low performance level, the results shown in Table 1 indicate that the CNN layer, at least in one of the cases, seems to have at least some positive effect on the overall performance. According to our theoretical considerations, these gains cannot stem from leveraging correlations in the embeddings as claimed in the papers, but might be merely the result of adding a neural network layer to the pretraining model, which acts as a universal

[7] This is appropriate because the optimization problem is unchanged and the train-test split remains identical.

[8] Due to the leave-one-out procedure, other metrics like Recall, Precision and F1 are linearly correlated to HR.

[9] See also [11, 36] on related problems of reproducing reported gains in the recommender systems literature.

Algorithm	Mode	Ablation Study 1		Ablation Study 2	
		NDCG	HR	NDCG	HR
ConvNCF	full	0.1623 ± 0.0008	0.3052 ± 0.0019	0.1623 ± 0.0008	0.3052 ± 0.0019
ConvNCF	element-wise	0.1622 ± 0.0008	0.3051 ± 0.0016	0.1632 ± 0.0012	0.3068 ± 0.0015
ConvNCF	correlations	0.0193 ± 0.0076	0.0403 ± 0.0150	0.1522 ± 0.0013	0.2900 ± 0.0020
CFM	full	0.1730 ± 0.0398	0.3155 ± 0.0724	0.1730 ± 0.0398	0.3155 ± 0.0724
CFM	element-wise	0.1730 ± 0.0398	0.3155 ± 0.0724	0.1805 ± 0.0034	0.3292 ± 0.0062
CFM	correlations	0.0015 ± 0.0003	0.0032 ± 0.0008	0.0011 ± 0.0001	0.0019 ± 0.0002
CoupledCF	full	0.5272 ± 0.0491	0.7865 ± 0.0470	0.5272 ± 0.0491	0.7865 ± 0.0470
CoupledCF	element-wise	0.5404 ± 0.0631	0.7744 ± 0.0994	**0.5763 ± 0.0059**	**0.8243 ± 0.0071**
CoupledCF	correlations	0.5137 ± 0.0903	0.7822 ± 0.0659	0.5503 ± 0.0343	0.7978 ± 0.0391

Table 2: Results of the two ablation studies. *Ablation Study 1* evaluates the contribution of each component of the interaction map to a model trained on the full map. *Ablation Study 2* evaluates the contribution of training on different parts of the interaction map. Significant improvements over the full map are printed in bold.

approximator. We conducted two types of ablation studies to further investigate this question.

Ablation Study 1. In order to measure how much the CNN model has learned to represent the embeddings correlation, we designed a novel type of ablation study, which was not part of the original papers. We started from the models previously trained on the full interaction map, but we computed the recommendations using only certain interaction map components. For CoupledCF, which does not use pretrained embeddings, we trained and evaluated the model on 20 random train-test splits.

Remember that the correlations in the embeddings are represented by the elements of the interaction map (Equation 1) that are not on the main diagonal. The following configurations were tested:

- *full* : This corresponds to the original setting.
- *element-wise*: Only the element-wise products (i.e., main diagonal elements) are considered.

$$e_{x,y} = \begin{cases} p_{u,x} \cdot q_{i,y} & \text{if } x = y \\ 0 & \text{otherwise.} \end{cases}$$

- *correlations*: Only the embeddings correlations (i.e., off-diagonal elements) are used.

$$e_{x,y} = \begin{cases} p_{u,x} \cdot q_{i,y} & \text{if } x \neq y \\ 0 & \text{otherwise.} \end{cases}$$

The results of this ablation study are reported in Table 2. They show that there is no statistical difference[10] between the models evaluated using the full interaction map and when only using the element-wise product (diagonal elements).

In other words, the off-diagonal correlation elements are not contributing anything to the overall performance. Interestingly, when the recommendations are computed using only the embeddings correlation, the observed results for ConvNCF and CFM are lower by at least an order of magnitude. This suggests both models have in some ways *learned to ignore the embedding correlations*. For CoupledCF, the accuracy obtained with the embeddings correlation is similar to the full interaction map. Therefore, although the model has learned to use the embeddings correlation, this did not prove to be beneficial for the model's quality. Overall, the results clearly indicate that the convolutional models are not learning to represent the embeddings correlation when these are pre-trained (i.e., ConvNCF and CFM). They also do not benefit from them even when they are learnable (CoupledCF), which is in direct contradiction to what was stated in the original articles.

Remember that in the original articles a similar ablation study was not present. The contribution of the embeddings correlation to the convolution model was therefore never directly measured.

Ablation Study 2. In *Ablation Study 1*, we observed that training the model on the full interaction map resulted in the embeddings correlation not contributing to improve the performance over the simple element-wise product.

While in *Ablation Study 1* we trained the model on the full interaction map, in *Ablation Study 2*, we isolate the different components of the interaction map at an earlier stage, i.e., during *the training phase*. Therefore, we do not train the network on the full map as in *Ablation Study 1*, but instead we train the model on specific components of the map.

In this new experiment, different models are therefore trained from scratch, using only the interaction map component associated with a given configuration (i.e., *full-map*, *element-wise*, *correlations*).[11] As a result, a model trained only on the element-wise product will never observe embeddings correlations and vice versa. This allows us to measure how much of each component the convolution algorithm learns to model when the other is not present.

The results of *Ablation Study 2* are also reported in Table 2. Since the training data (i.e., interaction map) fed to the CNN in the two

[10] The statistical significance of the difference between the observed results with α=0.05 was verified using a paired t-test if the values were normally distributed; and a Wilcoxon signed-rank test otherwise. To assess if the values of the metrics are normally distributed we used both Shapiro-Wilk and Kolmogorov-Smirnov tests.

[11] As discussed by Rendle et al. [35] in the element-wise configuration it should be trivial for a simple CNN to learn the dot product of the embeddings, since the dot product is equal to the diagonal of the outer product.

ablation studies are different, it is expected that the absolute values of the measurements are different. However, we can again observe that the results obtained when training the convolution model on the full interaction map and on the element-wise product are not different to a statistically significant extent for ConvNCF and CFM. The convolution operation is therefore not leveraging the embedding correlations in any effective way. As a result, these correlations can be discarded entirely during the training phase without degrading the model's performance. Remarkably, for CoupledCF we can observe that training the model on the element-wise product alone results in significantly better results than when using the full map. Remember that CoupledCF does not use pretrained embeddings but learns them during the training process, while ConvNCF and CFM, use pre-trained embeddings instead. Our result indicate that the additional parameter space provided by the learnable embedding correlations even seems to introduce noise, effectively harming the model quality.

Interestingly and differently from *Ablation Study 1*, we can see for ConvNCF and CoupledCF that training the convolution on the embeddings correlation elements alone yields results that are very close to those obtained when using the full interaction map. This suggests that the pretrained embeddings in ConvNCF do indeed carry some information that can, to an extent, be modeled. Similarly, CoupledCF can learn some correlations, although with a rather high standard deviation. However, the CNN models are, as demonstrated through *Ablation Study 1*, not able to leverage correlations to improve the accuracy obtained on the element-wise product alone.

There may be different reasons for this. First, it might be that the convolution on the full map was only able to model information that was redundant and already captured by the element-wise model. An alternative explanation is that the CNN model did not succeed in hybridizing these two pieces of information in a synergistic way, i.e., it only learned to select the best-performing or the less noisy one.

4.5 Comparison with Non-Neural Baselines

In all original papers investigated here, the claim is made that the proposed CNN-based algorithm is able to outperform the state-of-the-art. This claim is also used as demonstration that all models are effectively able to leverage embeddings correlations, which, as we showed before, is not the case. Recent research has found several instances of works where similar claims were possible only due to methodological issues such as the choice of weak baselines, the lack of proper optimization of the baselines, or information leakage from the test data [11, 25, 36]. In particular, two of the algorithms we analyze here (ConvNCF and CoupledCF) have been reported in [10] to be not competitive against simple baselines. We could replicate the performance results reported in [10] for ConvNCF and CoupledCF. Furthermore, we have conducted additional experiments of the same form for CFM, for which such an analysis was missing so far.

Like in Ferrari Dacrema et al. [10, 11], we compared all CNN-based algorithms to the same set of known non-neural, adequately optimized baseline algorithms. We used the evaluation framework shared by [10, 11], and extended the framework with an implementation of the CFM method which was made publicly available by the authors of the method.[12]

4.5.1 Baseline algorithms. The baseline algorithms we report here are a subset of those used in [10], and they represent algorithms of different families.

TopPopular A non-personalized model recommending the most popular items.

ItemKNN An item-based nearest neighbor model [39] using cosine similarity and shrinkage.

UserKNN A user-based nearest neighbor model [37] using cosine similarity and shrinkage.

$P^3\alpha$ A graph-based model implementing a random-walk between user and item nodes [5]. The method is equivalent to a KNN item-based CF algorithm, with the similarity matrix being computed as the dot-product of the probability vectors.

$RP^3\beta$ A version of $P^3\alpha$ which involves a reranking step [32].

PureSVD A matrix factorization approach based on the traditional SVD decomposition [7].

Sparse Linear Models (SLIM) An item-based recommendation model that learns the similarity between items via linear regression [31]. In our work, we use the more scalable variant proposed in [23].

Implicit Alternating Least Squares (iALS) A matrix factorization model for implicit feedback datasets proposed in [16]. In the iALS method, the implicit feedback signals are transformed into confidence values.

4.5.2 Hyperparameter optimization. In order to optimize the hyperparameters of the baseline methods we create a validation split from the train data, by applying the same splitting procedure that was used to create the test data. We use a Bayesian search [1, 12, 15], available as part of the Scikit-Optimize[13] package, exploring 50 hyperparameter configurations, with the first 15 used as an initial random initialization. Once hyperparameter values were found that optimize the recommendation accuracy on the validation data, we train the model again on the union of train and validation data and report the recommendation accuracy on the test data. Each baseline algorithm has a different set of hyperparameters. The complete list of these parameters as well as their range and distribution is the same as in reported in [10].

4.5.3 Results. The result of this comparison against simple baselines can be observed in Table 3 (ConvNCF), Table 4 (CoupledCF) and Table 5 (CFM). For all of these algorithms it was possible to reproduce both the experimental setting (i.e., the source code developed by the original authors and at least one dataset were available) and the numerical results reported in the original paper.

However, as it is possible to observe, we could not confirm that any of the algorithms is able to outperform the state-of-the-art.

- ConvNCF on the Yelp dataset is outperformed by all baselines, sometimes by more than 10%. Similar observations

[12]The results obtained for ConvNCF and CoupledCF refer to the same train-test split and hyperparameter setting and are, therefore, perfectly reproducing the results reported in [10].
[13]https://scikit-optimize.github.io/

were made for the Gowalla dataset, which served as a second dataset in the original paper.
- CoupledCF on Movielens1M is able to achieve recommendation accuracy results that are competitive with neighborhood-based and graph-based baselines. It is however not competitive with known techniques based on matrix factorization or linear regression. The results are similar for the Tafeng dataset used in the original paper.
- Lastly, CFM achieves surprisingly poor recommendation accuracy, sometimes only reaching half the level of the baselines for the Last.fm dataset. For the second dataset that was used to evaluate CFM in [43], we could not reproduce the original results because the code shared by the authors did not execute correctly[14].

Overall, our results indicate that the analyzed CNN-based algorithms, despite their computational complexity[15], are actually not able to outperform comparably simple and long-known baselines and to advance the state-of-the-art. While the use of an additional CNN layer may have some positive effects in some cases, these effects are *(i)* not the result of CNNs capturing correlations in the user-item interaction maps, *(ii)* not sufficient to raise the performance level of the CNN-based methods over the state-of-the-art.

Table 3: Experimental results for ConvNCF for the Yelp dataset.

	@5		@10		@20	
	HR	NDCG	HR	NDCG	HR	NDCG
TopPopular	0.0817	0.0538	0.1200	0.0661	0.1751	0.0799
UserKNN CF	**0.2068**	**0.1355**	**0.3126**	**0.1695**	0.4401	**0.2017**
ItemKNN CF	**0.2521**	**0.1686**	**0.3669**	**0.2056**	**0.4974**	**0.2385**
$P^3\alpha$	**0.2146**	**0.1395**	**0.3211**	**0.1737**	0.4442	**0.2049**
$RP^3\beta$	**0.2202**	**0.1431**	**0.3323**	**0.1793**	**0.4667**	**0.2132**
SLIM	**0.2330**	**0.1535**	**0.3475**	**0.1904**	**0.4799**	**0.2238**
PureSVD	**0.2011**	**0.1307**	0.3002	**0.1626**	0.4238	0.1938
iALS	**0.2048**	**0.1348**	**0.3080**	**0.1680**	0.4319	**0.1993**
ConvNCF	0.1947	0.1250	0.3059	0.1608	0.4446	0.1957

5 CONCLUSIONS

In this work, we analyzed recently published articles using CNNs on the interaction maps obtained from user and item embeddings. We argued that the original articles lacked a proper discussion on two crucial claims they made: *(i)* the analogy of embeddings and images, and *(ii)* the ability of CNNs to model embeddings correlations.

Our work has shown both through theoretical considerations and through empirically studies that embeddings do not share the topological properties of images. The use of CNNs is therefore not well justified since the embeddings interaction map does not exhibit feature locality and translation invariance, hence it is not analogous to an image. Moreover, we have shown that CNNs, as opposed to what was claimed in the original articles, are insensitive to the embeddings correlation and fail to improve over a model only using the element-wise product. Furthermore, we have compared the CNN based algorithms against a set of established and well optimized non-neural baselines. This was done using the same experimental setup as reported in the original papers. We could show that the proposed complex CNN algorithms were not able to outperform the state-of-the art.

Overall, while we do not argue that convolution cannot be applied on embeddings, we stress that a deeper understating and theoretical analyses of the semantics that new approaches are claimed to leverage are essential to obtain reliable progress in this field. Similarly, claims regarding the improved modelling capacity of an algorithm cannot be based simply upon its ability to outperform a set of baseline algorithms and datasets whose choice is not well justified. Instead, these aspects should be directly verified via specifically designed experiments.

Ultimately, our work also puts forward a new research question, which previously did not receive much attention. Specifically, the question is how to create an interaction map that captures potentially existing correlations in the data in its topology, such that CNNs can be successfully leveraged on these embeddings interaction maps.

Table 4: Experimental results for CoupledCF for the MovieLens1M dataset.

	@ 1		@ 5		@ 10	
	HR	NDCG	HR	NDCG	HR	NDCG
TopPopular	0.1593	0.1593	0.4217	0.2936	0.5813	0.3451
UserKNN CF	0.3540	0.3540	0.6884	0.5324	0.8060	0.5704
ItemKNN CF	0.3305	0.3305	0.6682	0.5080	0.7940	0.5488
$P^3\alpha$	0.3316	0.3316	0.6543	0.5031	0.7687	0.5402
$RP^3\beta$	0.3464	0.3464	0.6743	0.5198	0.7959	0.5591
SLIM	**0.3906**	**0.3906**	**0.7116**	**0.5625**	**0.8315**	**0.6014**
PureSVD	**0.3735**	**0.3735**	**0.7088**	**0.5522**	0.8132	**0.5861**
iALS	**0.3816**	**0.3816**	**0.7121**	**0.5581**	0.8200	**0.5933**
CoupledCF	0.3522	0.3522	0.7018	0.5374	0.8247	0.5775

Table 5: Experimental results for CFM for the Last.fm dataset.

	@5		@10		@20	
	HR	NDCG	HR	NDCG	HR	NDCG
TopPopular	0.0016	0.0009	0.0023	0.0011	0.0033	0.0014
UserKNN CF	**0.5964**	**0.4527**	**0.6715**	**0.4773**	**0.7032**	**0.4855**
ItemKNN CF	**0.5975**	**0.4425**	**0.6776**	**0.4689**	**0.7070**	**0.4764**
$P^3\alpha$	**0.6327**	**0.4929**	**0.6744**	**0.5066**	**0.7014**	**0.5135**
$RP^3\beta$	**0.5896**	**0.4458**	**0.6756**	**0.4739**	**0.7071**	**0.4821**
SLIM	**0.6674**	**0.5169**	**0.6972**	**0.5267**	**0.7102**	**0.5300**
PureSVD	**0.4026**	**0.3117**	**0.4891**	**0.3397**	**0.5652**	**0.3590**
iALS	**0.6110**	**0.4811**	**0.6735**	**0.5017**	**0.7033**	**0.5093**
CFM	0.2241	0.1485	0.3338	0.1839	0.4661	0.2173

[14] We contacted all the authors to resolve the issue, but without success.
[15] Even on high-end GPUs, the computation time need to fit the CNN models including early stopping on a typical dataset is about 8 hours for CoupledCF, 17 hours for CFM and 24 hours for ConvNCF.

REFERENCES

[1] Sebastiano Antenucci, Simone Boglio, Emanuele Chioso, Ervin Dervishaj, Shuwen Kang, Tommaso Scarlatti, and Maurizio Ferrari Dacrema. 2018. Artist-driven Layering and User's Behaviour Impact on Recommendations in a Playlist Continuation Scenario. In *Recommender Systems Challenge Workshop at the 12th ACM Conference on Recommender Systems*. 4:1–4:6.

[2] Timothy G. Armstrong, Alistair Moffat, William Webber, and Justin Zobel. 2009. Improvements That Don't Add Up: Ad-hoc Retrieval Results Since 1998. In *Proceedings of CIKM '09*. 601–610.

[3] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2018. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv:1803.01271*.

[4] James Bennett and Stan Lanning. 2007. The Netflix Prize. In *KDD Cup Workshop 2007*.

[5] Colin Cooper, Sang Hyuk Lee, Tomasz Radzik, and Yiannis Siantos. 2014. Random walks in recommender systems: exact computation and simulations. In *Proceedings of WWW '14*. 811–816.

[6] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of RecSys '16*. 191–198.

[7] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of Recommender Algorithms on Top-n Recommendation Tasks. In *Proceedings of RecSys '10*. 39–46.

[8] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of NIPS '16*. 3844–3852.

[9] Xiaoyu Du, Xiangnan He, Fajie Yuan, Jinhui Tang, Zhiguang Qin, and Tat-Seng Chua. 2019. Modeling Embedding Dimension Correlations via Convolutional Neural Collaborative Filtering. *ACM TOIS* 37, 4.

[10] Maurizio Ferrari Dacrema, Simone Boglio, Paolo Cremonesi, and Dietmar Jannach. 2019. A Troubling Analysis of Reproducibility and Progress in Recommender Systems Research. *arXiv:1911.07698*.

[11] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. *Proceedings of RecSys '19*.

[12] Antonino Freno, Martin Saveski, Rodolphe Jenatton, and Cédric Archambeau. 2015. One-Pass Ranking Models for Low-Latency Product Recommendations. In *Proceedings of KDD '15*. 1789–1798. http://doi.acm.org/10.1145/2783258.2788579

[13] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer Product-based Neural Collaborative Filtering. In *Proceedings of IJCAI '18*. 2227–2233.

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of WWW '17*. 173–182.

[15] José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. 2014. Predictive Entropy Search for Efficient Global Optimization of Black-box Functions. In *Proceedings of NIPS '14*. 918–926.

[16] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets.. In *Proceedings of ICDM '08*. 263–272.

[17] Junyang Jiang, Deqing Yang, Yanghua Xiao, and Chenlu Shen. 2019. Convolutional Gaussian Embeddings for Personalized Recommendation with Uncertainty. In *Proceedings of IJCAI '19*. 2642–2648.

[18] Christopher C Johnson. 2014. Logistic matrix factorization for implicit feedback data. In *Proceedings of NIPS '14*, Vol. 27.

[19] Yehuda Koren and Robert Bell. 2011. *Advances in Collaborative Filtering*. 145–186.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of NIPS '12*. 1097–1105.

[21] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE*, 2278–2324.

[22] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. 2009. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of ICML '09*. 609–616.

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

[24] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *Proceedings of WWW '18*. 689–698.

[25] Jimmy Lin. 2019. The Neural Hype, Justified! A Recantation. *SIGIR Forum* 53, 2 (2019), 88–93.

[26] Zachary C. Lipton and Jacob Steinhardt. 2018. Troubling Trends in Machine Learning Scholarship. In *Proceedings of ICML '18: The Debates*. arXiv:1807.03341

[27] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of CVPR '13*. 3431–3440.

[28] Malte Ludewig and Dietmar Jannach. 2018. Evaluation of Session-based Recommendation Algorithms. *UMUAI* 28, 4–5, 331–390.

[29] Malte Ludewig, Noemi Mauro, Sara Latifi, and Dietmar Jannach. 2019. Performance Comparison of Neural and Non-Neural Approaches to Session-based Recommendation. In *Proceedings of RecSys '19*.

[30] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. 2018. Statistical and Machine Learning forecasting methods: Concerns and ways forward. *PloS one* 13. Issue 3.

[31] Xia Ning and George Karypis. 2011. SLIM: Sparse linear methods for top-n recommender systems. In *Proceedings of ICDM '11*. 497–506.

[32] Bibek Paudel, Fabian Christoffel, Chris Newell, and Abraham Bernstein. 2017. Updatable, Accurate, Diverse, and Scalable Recommendations for Interactive Applications. *ACM TiiS* 7, 1.

[33] Steffen Rendle. 2010. Factorization Machines. In *Proceedings of ICDM '10*. 995–1000.

[34] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI '09*. 452–461.

[35] Steffen Rendle, Walid Krichene, Li Zhang, and John Anderson. 2020. Neural Collaborative Filtering vs. Matrix Factorization Revisited. In *Proceedings of RecSys '20*.

[36] Steffen Rendle, Li Zhang, and Yehuda Koren. 2019. On the Difficulty of Evaluating Baselines: A Study on Recommender Systems. *CoRR* abs/1905.01395. http://arxiv.org/abs/1905.01395

[37] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of CSCW '94*. 175–186.

[38] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. In *Proceedings of NIPS '17*. 3856–3866.

[39] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *Proceedings of WWW '01*. 285–295.

[40] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of WSDM '18*. 565–573.

[41] Srinivas C Turaga, Joseph F Murray, Viren Jain, Fabian Roth, Moritz Helmstaedter, Kevin Briggman, Winfried Denk, and H Sebastian Seung. 2010. Convolutional networks can learn to generate affinity graphs for image segmentation. *Neural Computation* 22, 511–538. Issue 2.

[42] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Proceedings of NIPS '13*. 2643–2651.

[43] Xin Xin, Bo Chen, Xiangnan He, Dong Wang, Yue Ding, and Joemon Jose. 2019. CFM: Convolutional Factorization Machines for Context-Aware Recommendation. In *Proceedings of IJCAI '19*. 3926–3932.

[44] Wei Yang, Kuang Lu, Peilin Yang, and Jimmy Lin. 2019. Critically Examining the Neural Hype: Weak Baselines and the Additivity of Effectiveness Gains from Neural Ranking Models. In *Proceedings of SIGIR '19*. 1129–1132.

[45] Fisher Yu and Vladlen Koltun. 2015. Multi-Scale Context Aggregation by Dilated Convolutions. *CoRR* abs/1511.07122. https://arxiv.org/abs/1511.07122

[46] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M. Jose, and Xiangnan He. 2019. A Simple Convolutional Generative Network for Next Item Recommendation. In *Proceedings of WSDM '19*. 582–590.

[47] Bangzuo Zhang, Haobo Zhang, Xiaoxin Sun, Guozhong Feng, and Chunguang He. 2018. Integrating an Attention Mechanism and Convolution Collaborative Filtering for Document Context-Aware Rating Prediction. *IEEE Access* 7, 3826–3835.

[48] Quangui Zhang, Longbing Cao, Chengzhang Zhu, Zhiqiang Li, and Jinguang Sun. 2018. CoupledCF: Learning Explicit and Implicit User-item Couplings in Recommendation for Deep Collaborative Filtering. In *Proceedings of IJCAI '18*. 3662–3668.

[49] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep Learning Based Recommender System: A Survey and New Perspectives. *Comput. Surveys*, Article 5, 5:1–5:38 pages.