July 2021

You'll like the sound of this: Deep Item-based Collaborative Filtering on songs

A 7.5 ECTS thesis submitted to the Faculty of Humanities at

Utrecht University

in partial fulfillment of the requirements for the degree of

Bachelor of Science in Artificial Intelligence

Author
Sem Yedema

Supervisor and examiner
Dr. R.A. Mercuur

Secundary examiner
Dr. B.G. Rin

Abstract

Recommender systems are becoming increasingly important in the growing online society. DeepICF and DeepICF+a are recommender system algorithms which have been claimed to provide more suitable recommendations than alternative algorithms. However, DeepICF and DeepICF+a have only been tested on movies and pictures, using HR and NDGC as evaluation metrics. This paper aims to explore the generalizability of both algorithms, using the same evaluation metrics, following several steps. First, this paper reproduces original experiments with DeepICF and DeepICF+a and shows that the accuracy of both algorithms is reduced through the process of reproduction. Then, the differences between explicit and implicit ratings, as well as the effects of pre-training, are investigated. The use of implicit ratings instead of explicit ratings has no negative influence on the performance of DeepICF and DeepICF+a, while the absence of pre-training data does have a negative impact on their accuracy. Finally, DeepICF and DeepICF+a are applied to the Million Song Dataset in an attempt to examine the generalizability of these algorithms, with FISM as baseline algorithm. Compared to the originally used data sets, all three algorithm returned less accurate recommendation lists on the Million Song Dataset. The used FISM implementation produced malfunctioning output on this data set, so it remains unproven that DeepICF and DeepICF+a outperform FISM on the Million Song Dataset. Still, it is found that DeepICF+a is more sensitive to parameter settings than DeepICF, and yields worse result when not adjusted properly. For practical implementations, DeepICF is therefore the better option of the two—in fact, it has yet to be proven inferior to any other recommender system algorithm.

Acknowledgments

Rijk, I am thankful for your guidance, feedback and helpfulness.

Ben, thank you for your time, attention and examination.

Jeroen, I appreciate our conversation, which provided me with much insight.

Sara, you are the best study partner one could wish for. I thank you for that.

Contents

1 Introduction

The internet contained around 33 zettabytes[1] worth of data in 2018, a size which will reach 175 zettabytes by 2025, while never ceasing to expand (Rydning et al., 2018). Hence, data filtering is becoming increasingly important for gathering relevant information. Recommender systems can be used for this: given some input data about a user, they predict to what extent a user would be interested in some piece of data or other (Ricci et al., 2011). They are widely used in online stores and entertainment platforms, because they improve the user's decision-making process and quality (Pathak et al., 2010) as well as their satisfaction and persuasion (Nanou et al., 2010), while also enhancing revenues by increasing sales (Pu et al., 2011; Dias et al., 2008). Thus, the optimization of recommender systems performance is beneficial for both internet users and companies.

Most current recommender systems are based on collaborative filtering (CF), a technique that (1) predicts the preference for items—i.e. pieces of data—for individual users by incorporating other users' preferences for those items; and (2) subsequently recommends a selection of items accordingly (Ekstrand et al., 2011). Collaborative filtering can be categorized into two main approaches: user-based CF (UCF) and item-based CF (ICF).[2] Both algorithms can be used successfully, but recently, ICF algorithms have been showing more accurate prediction results than UCF algorithms (Xue et al., 2019). Moreover, ICF is generally more easily scaled and less intensive, computationally speaking (Sarwar et al., 2001; Lü et al., 2012). Therefore, although hybrid approaches have been proposed (Thakkar et al., 2019), this paper focuses primarily on item-based collaborative filtering.

Previously, ICF algorithms calculated the similarity between items through statistical measurements (Sarwar et al., 2001). This method is impractical, because of the amount of manual tuning it requires to perform well (Xue et al., 2019). More recently, with the rise of machine learning, algorithms have been developed that learn the item-item relations directly from data (Ning & Karypis, 2011; Kabbur et al., 2013; Christakopoulou & Karypis, 2014; Wu et al., 2016). However, these ICF approaches have been criticized by Xue et al., for being unable to effectively capture higher-order or nonlinear relations between items.[3]

That is why Xue et al. (2019) developed an algorithm dubbed Deep Item-based Collaborative Filtering (DeepICF). DeepICF implements deep learning to discover those higher-order and nonlinear item-item relations. Moreover, their paper presented a variation of this algorithm, called DeepICF+a.[4] Henceforth, 'DeepICF(+a)' shall be used as an abbreviation for 'DeepICF and DeepICF+a'. They trained and tested DeepICF(+a) on two data sets. One is called MovieLens 1 Million (MovieLens for short) and contains movies and accompanying ratings provided per individual user. The other consists of photos from Pinterest with implicit ratings and shall simply be called Pinterest from now on. In most cases, implicit ratings are binary: 1 (if the user has interacted with this item) or 0 (if he/she has not). Subsequently, Xue et al. tested

[1] One zettabyte is equal to one trillion (i.e. 1,000,000,000,000) gigabytes.
[2] In Section 2.1, both approaches are explained and compared.
[3] See Section 2.2 for an explanation of these types of item-item relations.
[4] Section 2.4 contains descriptions of DeepICF and DeepICF+a.

alternative ICF algorithms on these two data sets and evaluated their performance by calculating the Hit Rate (HR) and Normalized Discounted Cumulative Gain (NDCG) for each algorithm including their own.[5] Finally, they concluded from the results that their algorithm outperformed the others.

However, DeepICF(+a) were only tested on movies and pictures. It is still unknown how well the algorithms work in other domains, such as that of songs, for example. An available data set in this domain–and the data set of choice for this paper—is the Million Song Dataset (MSD). On one hand, if DeepICF(+a) outperform alternative recommender systems in this domain, too, it would be further confirmation of their superiority, as well as an indication of their generalizability. On the other hand, if they deliver subpar results, the algorithms' range of applicability is proven to be limited. Either way, the outcome would contribute to the development of recommender systems by determining to what extent DeepICF(+a) can be generalized to a data set in the domain of songs.

Before this can be determined, however, other subjects involving DeepICF(+a) must be treated. Thus, this paper answers the following research questions:

RQ1 What HR and NDCG do DeepICF(+a) and FISM obtain in reproduction of—and in comparison with—the original experiments by Xue et al. (2019), with default parameters?

RQ2 What effect does the conversion of MovieLens from explicit to implicit (binary) ratings have on the HR and NDCG attained by DeepICF(+a)?

RQ3 What effect does pre-training have on the HR and NDCG attained by DeepICF(+a) when applied to MovieLens?

RQ4 How do the HR and NDCG achieved by DeepICF(+a) on MSD relate to the HR and NDCG achieved by DeepICF(+a) on MovieLens and Pinterest?

RQ5 How do the HR and NDCG achieved by DeepICF(+a) on MSD relate to the HR and NDCG achieved by FISM on MSD?

FISM (Factored Item Similarity Models) is used as the baseline algorithm, because it is "the state-of-the-art item-based CF method", according to Xue et al. (2019, p. 14) themselves.

The first three research questions serve to provide context to the answers resulting from the last two research questions, so that it can be interpreted what those answers entail. First, relevant extensive theory is provided in Section 2. Section 3 follows with a description of the processing the data sets go through, and of their structure. Then, RQ1 is treated in Section 4, where the reproduced results are found to be lower than the results obtained by Xue et al. (2019). Subsequently, Section 5 investigates RQ2 and RQ3. It shows that the use of implicit ratings has no negative influence on the results. Furthermore, that section finds that pre-training positively impacts the accuracy of DeepICF(+a). RQ4 and RQ5 are explored in Section 6. It is discovered that DeepICF(+a) perform much less accurately on MSD than on MovieLens and

[5] Refer to Section 2.3 for more information on these evaluation metrics.

Pinterest, but still higher than FISM—although the used FISM implementation does not function properly when applied to MSD. Therefore, while this paper provides much insight into DeepICF(+a) and related matters—discussed in Section 7—it cannot definitively be affirmed nor negated that DeepICF(+a) outperform FISM. This conclusion is part of Section 8, which finalizes this paper.

1.1 Relevance to artificial intelligence

This paper analyses DeepICF(+a). As mentioned, these algorithms make use of a neural network to discover relations between items (Xue et al., 2019). Specifically, they use deep learning, which is a form of machine learning (Goodfellow et al., 2016). Machine learning, in turn, is a subset of artificial intelligence (Zhang, 2020). Thus, this paper is relevant to artificial intelligence, because it provides insight into the generalizability of a proposed state-of-the-art algorithm that uses deep learning.

2 Background

This section explains some concepts which are important to understand the content of this paper. Subjects include: UCF and ICF (Section 2.1); types of item-item relations (Section 2.2); used evaluation metrics (Section 2.3); and DeepICF(+a) and FISM (Section 2.4).

2.1 User-based and Item-based Collaborative Filtering

In UCF, a user's preference for items is predicted by determining the k most similar users, before recommending the items that received the greatest appreciation from the k similar users (Su & Khoshgoftaar, 2009). In ICF, it is not the user-user similarity that is identified. Instead, the similarity between item pairs is computed, e.g. by examining what percentage of users who expressed preference for item 'X' also expressed preference for item 'Y'. Principally, these relations are more static than in UCF, because new data does not require updating the similarity between all users that interacted with this item. Rather, only the relevant item pairs should have their similarity updated (Xue et al., 2019). This makes ICF generally more scalable and less computationally intensive than UCF (Sarwar et al., 2001; Lü et al., 2012). For that reason, Xue et al. (2019) pursue ICF in their effort to create the most accurate recommender system algorithms with DeepICF(+a).

2.2 Item-item relations

2.2.1 Second-order and higher-order relations

Second-order relations are also known as pairwise relations (Xue et al., 2019), which gives an indication of their nature: a second-order relation is a connection between two items that can be described in terms of pairs (Agarwal, 2006). In contrast, higher-order relations have more than just two influential factors, according to Christakopoulou and Karypis (2014) and Xue et al.. Christakopoulou and Karypis

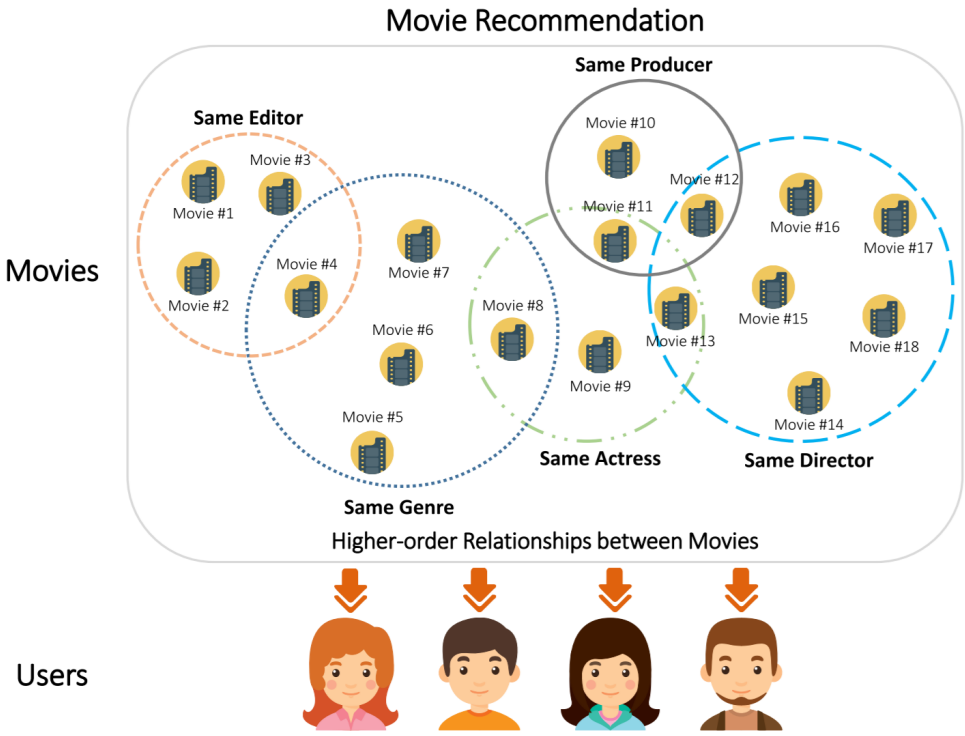


Figure 1: An illustrative example of higher-order relations between items. Retrieved from Xue et al. (2019).

give the example of shopping for the ingredients of a recipe. These items are probably not directly similar, but only through the connection of belonging to the same recipe—a third entity. Xue et al. provide a similar example with two movies whose relation is that they share a director or actress—also a third entity (Figure 1).

2.2.2 Linear and nonlinear relations

Linear relations are such, that a change in one value is directly proportional to one or more other values. To clarify, picture a graph that portrays a linear relation and you will, quite literally, see a line. Nonlinear relations, in turn, are all relation which do not fit this criteria. A graph of a nonlinear relation may depict a curve or a more complex figure (*Nonlinear Relationships and Graphs without Numbers*, 2016). Xue et al. (2019) argue that are nonlinear relations between items are present and influential, but not taken advantage of properly by alternative ICF algorithms.

2.3 Evaluation metrics

2.3.1 Hit Ratio (HR)

HR indicates relatively how many times the correct prediction is included in the list of recommended items—this is called a 'hit'. It is calculated as follows (Christakopoulou & Karypis, 2014):

$$HR = \frac{\#\text{hits}}{\#\text{recommendation lists}}.$$

More specific notation of this evaluation metric is HR@k, where k is equal to the size of the provided recommendation lists. In this paper, $k = 10$ for all algorithms, so the used metric, HR@10, will be abbreviated to just HR.

To clarify, HR can range from 0 to 1 inclusive, and a higher HR generally means higher accuracy. Additionally, when providing random recommendation lists of length $k = 10$ in the data sets described in Section 3, which have 1 out of 100 positive interactions in the test set, the expected HR is 0.1.

It should be noted that HR may be considered not very informative, since it does not reflect a difference between when the correct prediction is in the first or last place of the recommendation list. Xue et al. (2019) therefore opted to use NDCG alongside this metric when evaluating DeepICF(+a).

2.3.2 Normalized Discounted Cumulative Gain (NDCG)

NDCG—or NDCG@k, analogously to HR—addresses above-mentioned limitation of HR and incorporates the correct item's place in the recommendation list (Busa-Fekete et al., 2012). In DeepICF(+a), Xue et al. (2019) implemented NDCG as follows:

$$NDCG_u = \begin{cases} \frac{\log(2)}{\log(i+2)} & \text{if } i < k \\ 0 & \text{otherwise} \end{cases},$$

where $NDCG_u$ denotes NDCG per user and i is equal to the correct item's rank (starting at 0). $NDCG_u$ averaged over all users results in the form of NDCG that is used by Xue et al. and this paper. Like HR, NDCG has a value between 0 and 1, where higher values indicate higher accuracy. Table 1 shows all values $NDCG_u$ can assume with recommendation list length $k = 10$.

2.4 DeepICF(+a) and FISM

This paper focuses on the application of DeepICF(+a) and FISM, and on the comparison between their performances. This section describes the algorithms on a conceptual level, which should be sufficient to understand their function for the scope of this paper. If you would like to learn more about their mathematical details notwithstanding the descriptions below, please refer to Xue et al. (2019) for DeepICF(+a), and to Kabbur et al. (2013) for FISM.

Figure 2 illustrates the architecture of the neural network utilized by DeepICF. The input layer and the embedding layer are used to convert the data in such a way that it produces vectors that can be processed by the neural network. Subsequently,

i	$NDCG_u$
0	1.0000
1	0.6309
2	0.5000
3	0.4307
4	0.3869
5	0.3562
6	0.3333
7	0.3155
8	0.3010
9	0.2891
≥ 10	0.0000

Table 1: Individual NDCG scores per item rank i with recommendation list length $k = 10$.

the pairwise interaction layer calculates the second-order (i.e., pairwise) similarities between couples of items. Next, the pooling layer converts all the vectors to the same size for further processing. Vectors may have different sizes, since not every user may have interacted with the same number of items. Then, in the deep interaction layers, higher-order relations are discovered. Finally, the output is a prediction score. The 10 items with the highest prediction score are then presented as the recommendation list.

DeepICF+a follows the same architecture as DeepICF in Figure 2. The only difference is that a hidden layer is added between the pairwise interaction layer and the pooling layer. This hidden layer functions as an attention mechanism: it prioritizes some item-item relations over others. Hence the name 'DeepICF+a', which is short for '**Deep I**tem-based **C**ollaborative **F**iltering with **a**ttention mechanism'.

The architecture of FISM is also very similar to that of DeepICF (Figure 2). A crucial difference, however, is that the deep interaction layers are not present in FISM. This is why FISM is incapable of detecting higher-order relations between items, according to Xue et al. (2019). Since the deep interaction layers are not implemented, FISM does not have a pooling layer, either.

3 Data set processing and structure

Three data sets are used in this paper: MovieLens,[6] Pinterest[6] and MSD.[7] Specifically, the Echo Nest Taste Profile subset of MSD is used. Note that these abbreviations are not used to denote the original unprocessed data sets, but rather the processed data sets as described below.

MovieLens and Pinterest have already been filtered by Xue et al. (2019) so that they only contain users with at least 20 different item interactions. Therefore, MSD

[6]MovieLens and Pinterest are downloadable via `https://github.com/linzh92/DeepICF`, along with the code for DeepICF(+a).

[7]MSD is available at `http://millionsongdataset.com/tasteprofile/`.

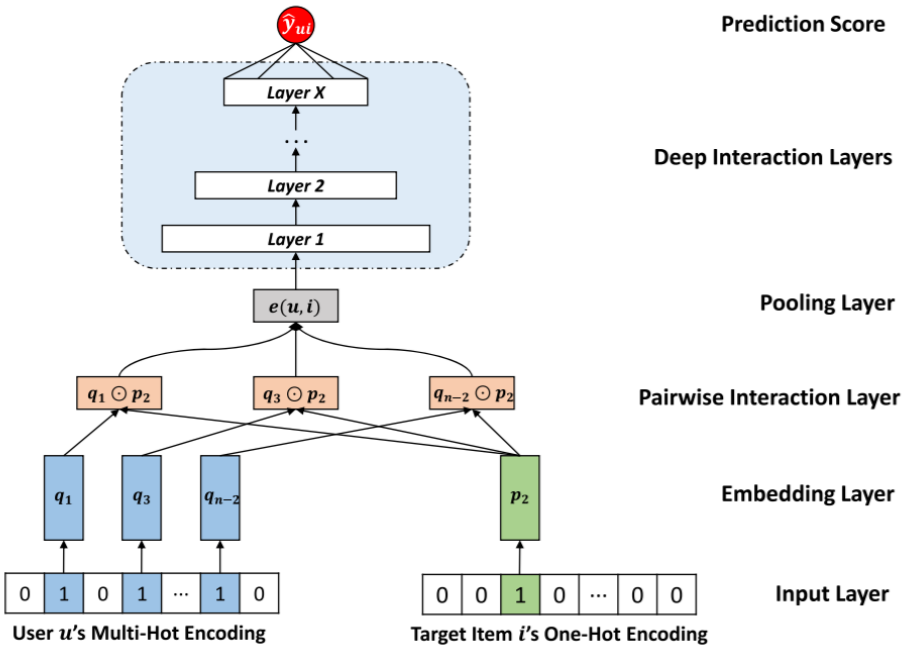


Figure 2: The neural network architecture of DeepICF. Retrieved from Xue et al. (2019). $\boldsymbol{q}$ and $\boldsymbol{p}$ are vectors, representing some user $\boldsymbol{u}$ and some item $\boldsymbol{i}$, respectively. $\boldsymbol{e}(\boldsymbol{u},\boldsymbol{i})$ is the pooling function over $\boldsymbol{u}$ and $\boldsymbol{i}$. $\hat{y}_{ui}$ denotes the prediction score given $\boldsymbol{u}$ and $\boldsymbol{i}$.

is filtered identically. However, MSD still contains over 40 million rows of data after this operation. Since this amount of data is impractical for this paper's research purpose, 1% of users is selected at random and only the corresponding data rows are stored.

All three data sets contain user IDs, item IDs, ratings and, when available, timestamps. The user IDs and item IDs (i.e., song IDs) in MSD are converted from alphanumeric to numeric in order to be compatible with DeepICF(+a). Furthermore, MSD does not contain ratings, but play counts per song per user, instead. These are converted to binary, implicit ratings. Thus, all *positive* interactions (i.e., user-item interactions that have taken place) have a rating of 1, like in Pinterest. MovieLens contains explicit ratings ranging from 1 to 5.

Subsequently, MovieLens, Pinterest and MSD are split into a train set and a test set: the latest interaction of every user is moved to the test set. The data that remains constitutes the train set. Each positive interaction in the test set is then accompanied by 99 randomly selected *negative* interactions—items with which that particular user has *not* interacted.

At this point, the structure of all three data sets is identical to how they are used by Xue et al. In Table 2, each data set's number of interactions; number of different users; number of different items; number of uniquely occurring items; and density is included.

Data set	#interactions	#users	#items	#unique-items	Density
MovieLens	1000209	6040	3706	114	0.0447
Pinterest	1500809	55187	9916	68	0.0027
MSD	437703	6611	100967	49395	0.0007

Table 2: Metadata of the data sets used in this paper.

4 Verifying the implementation of DeepICF(+a) and FISM

4.1 Motivation

This section recreates the experiments with DeepICF(+a) and FISM by Xue et al. (2019). The results are compared to those attained by them. If there is a discrepancy between the two sets of results (i.e., sets of HR and NDCG scores), this discrepancy should be taken into account when drawing conclusions from the results of this research. Furthermore, if this paper's recreated experiments with DeepICF(+a) deliver much lower HR and NDCG scores, it implies that the algorithms are actually quite unintuitive to implement—at least for this paper—even though Xue et al. argue that the implementation of DeepICF(+a) should be relatively intuitive, compared to some other ICF approaches.

Ultimately, this section seeks to answer RQ1: "What HR and NDCG do DeepICF(+a) obtain in reproduction of—and in comparison with—the original experiments by Xue et al. (2019), with default parameters?".

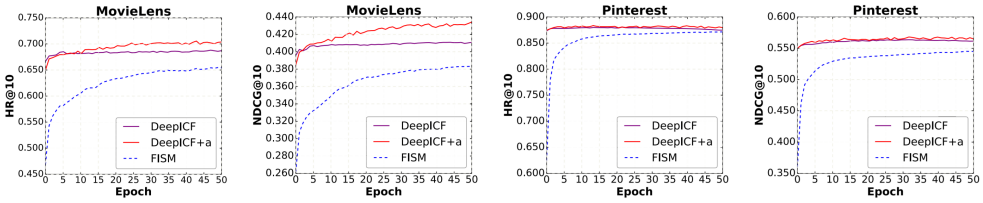
4.2 Methodology

DeepICF(+a) are applied to MovieLens and Pinterest, which were used as data sets by Xue et al. (2019).

The released files for DeepICF(+a) only contain pre-train data for MovieLens. Since this is not made explicit by Xue et al., it is assumed that their results on Pinterest were attained without the use of pre-training. Thus, in recreating these experiments, pre-training is only implemented in the application of DeepICF(+a) to MovieLens, and not Pinterest.

It is deducted from Figure 3, that there is no visible improvement in HR after 50 epochs. Therefore, DeepICF(+a) are set to terminate after this number has been reached.

Unfortunately, Xue et al. did not specify how they implemented FISM in their experiments. Moreover, the creators of FISM, Kabbur et al. (2013), did not provide



(a) MovieLens – HR@10 (b) MovieLens – NDCG@10 (c) Pinterest – HR@10 (d) Pinterest – NDCG@10

Figure 3: Testing performance of DeepICF(+a) and FISM as executed by and retrieved from Xue et al. (2019).

the code for their implementation of this algorithm in their paper, either. Hence, the most popular available implementation of FISM is adapted from GitHub and used for this paper.[8]

Copies of MovieLens and Pinterest are processed so that they are compatible with this particular FISM implementation. Moreover, since this implementation of FISM only works with binary ratings, MovieLens ratings are converted accordingly—Pinterest ratings were already binary. Due to this implementation's popularity, it is possible that Xue et al. also used it. Section 5 finds that this explicit–implicit rating conversion has no negative effect on HR and NDCG, so this conversion of MovieLens is disregarded.

Furthermore, the formula for NDCG (Section 2.3.2) is added to the FISM code, because this implementation originally only measured HR.

Aside from this, like DeepICF(+a), FISM is terminated when the HR improvement stagnates. For FISM, this is determined to be the case when the latest 5 HR scores are not higher than the best HR so far.

Any and all other parameters are not modified. These will assume their default values as provided by Xue et al. and S. Yu, the owner of the used FISM code: they have both experimented with different parameter settings and are expected to have set the best performing values as default parameters. Even if they have not, it is beyond the scope of this paper to engage in this experimentation.

The accuracy results of the algorithms are measured in HR and NDCG. Additionally, the percentage change from the original results to the reproduced results is calculated and used.

4.3 Results

The HR and NDCG scores resulting from the application of DeepICF(+a) and FISM to MovieLens and Pinterest, as obtained by Xue et al. (2019) and as reproduced, are shown in Table 3. Additionally, Table 3 contains the reproduced experimental results' change in HR and NDCG scores as compared to the results of Xue et al.

[8]This is available at `https://github.com/yushuai/FISM`.

Algorithm	Metric	MovieLens			Pinterest		
		Xue	Repr.	*change*	Xue	Repr.	*change*
DeepICF	HR	0.6881	0.6848	−0.5%	0.8806	0.8744	−0.7%
	NDCG	0.4113	0.4089	−0.6%	0.5631	0.5526	−1.9%
DeepICF+a	HR	0.7084	0.6773	−4.4%	0.8835	0.8253	−6.6%
	NDCG	0.4380	0.4045	−7.6%	0.5666	0.4970	−12.3%
FISM	HR	0.6685	0.6522	−2.4%	0.8763	0.8485	−3.2%
	NDCG	0.3954	0.3841	−2.9%	0.5529	0.5196	−6.0%

Table 3: DeepICF(+a) and FISM HR and NDCG scores as obtained by Xue et al. (2019) (denoted as 'Xue') and as obtained through reproduction (denoted as 'Repr.')

4.4 Discussion

Based on Table 3, it seems that the implementations of DeepICF(+a) and FISM used by this paper attained exclusively lower prediction accuracy than the implementations by Xue et al. (2019). This is likely due to the fact that mostly default values were used as algorithm parameters. Customizing parameter settings may have increased the accuracy of the reproduced experimental results, but this process was purposely avoided for the sake of this paper's conciseness. So, unless explicitly mentioned, default values have been used. Nevertheless, this decrease in accuracy should be taken into account when interpreting the results attained in Section 6.

Notably, all reproduced results from the experiments with DeepICF+a were lower than those from the experiments with DeepICF. Specifically, when comparing the reproduced DeepICF+a results to the reproduced DeepICF results, HR and NDCG decreased by 1% on MovieLens; HR decreased by 5% on Pinterest; and NDCG decreased by 10% on Pinterest. This is remarkable because all original results obtained by Xue et al. with DeepICF+a were higher than their results with DeepICF. In fact, with an average reduction in HR and NDCG of 5.5% and 10.0%, respectively, DeepICF+a results were most negatively affected through the process of reproduction. On average, compared to Xue et al.'s results, DeepICF HR and NDCG only decreased by 0.6% and 1.2%, respectively; FISM average HR and NDCG were 2.8% and 4.4% lower, respectively. This relatively large decrease even led to DeepICF+a being less accurate in its predictions on Pinterest than FISM. Therefore, besides the overall decrease in accuracy through reproduction, the negative impact on DeepICF+a results in particular must especially be considered in Section 6.[9]

In an explicit answer to RQ1 (Section 4.1): in this paper's reproductions, DeepICF(+a) and FISM obtain exclusively lower HR and NDCG in comparison with the original experiments by Xue et al. (2019), with default parameters.

[9]The standard deviation between runs performed by Xue et al. is not available, so it cannot be determined with certainty that these results are significant or not. Therefore, the focus lies on the direction of change and the relation between the results.

5 Effects of explicit–implicit rating conversion and pre-training

The aim of this section is to investigate the influence of two differences in algorithm implementations between the experiments performed by Xue et al. (2019) and those performed for this paper. First, in Section 5.1, it is investigated whether the use of implicit (i.e., binary) ratings over explicit (i.e., natural number) ratings in a data set has any impact on the accuracy of DeepICF(+a). This section answers RQ2. Subsequently, Section 5.2 researches the influence of pre-training on DeepICF(+a) to answer RQ3.

5.1 Explicit–implicit rating conversion effects

5.1.1 Motivation

As mentioned in Section 4, the FISM implementation used in this paper only works with data sets containing implicit ratings. Therefore, for a fair comparison between DeepICF(+a) and FISM on a new data set, this data set must contain implicit ratings, too. As such, it is useful to discover how DeepICF(+a) perform on implicitly rated items as opposed to explicitly rated items. For example, it could be that DeepICF(+a) perform better on ratings with natural numbers than on binary ratings, because the former contain inherently more information than the latter. Also, DeepICF(+a) could simply be better suited for either type of ratings. If, for instance, DeepICF(+a) provide much less accurate predictions on implicit ratings than on explicit ratings, this could imply that DeepICF(+a) and FISM should coexist: in data sets with implicit rankings, FISM would be preferred; and in data sets with explicit rankings, DeepICF(+a) would be more applicable.

Primarily, this section answers RQ2: "What effect does the conversion of MovieLens from explicit to implicit (binary) ratings have on the HR and NDCG attained by DeepICF(+a)?".

5.1.2 Methodology

To see how DeepICF(+a) are affected by the type of ranking within a data set, both algorithms are applied to MovieLens as provided by Xue et al. (2019). The results from the same run have been used in Section 4. Thereafter, a copy of MovieLens is made, where all user ratings of interacted items are converted to 1, with no further modifications. DeepICF(+a) are then also applied to this modified data set, which shall be called 'MovieLens-BI'.

Since no pre-training data is available for DeepICF(+a) on MovieLens-BI, it is also disabled for their applications on MovieLens.

Finally, the algorithms are terminated after 50 epochs; all other parameters are left to their default values; and HR and NDCG are used as evaluation metrics.

Algorithm	Metric	MovieLens	MovieLens-BI	*change*
DeepICF	HR	0.6565	0.6714	+2.3%
	NDCG	0.3876	0.3954	+2.0%
DeepICF+a	HR	0.6199	0.6197	−0.0%
	NDCG	0.3537	0.3584	+1.3%

Table 4: DeepICF(+a) HR and NDCG scores on MovieLens (with explicit ratings) and MovieLens-BI (with implicit ratings) without pre-training

5.1.3 Results

Table 4 shows the resulting HR and NDCG scores of the prediction accuracy of DeepICF(+a) when applied to MovieLens and MovieLens-BI without pre-training. Moreover, Table 4 presents the relative change in HR and NDCG when switching from MovieLens to MovieLens-BI.

5.1.4 Discussion

The results as shown in Table 4 indicate that DeepICF(+a) do not provide less accurate predictions on data sets with implicit ratings as opposed to data sets with explicit ratings. Actually, the binary format of MovieLens-BI seemed to have slightly improved their performance. The cause of this could be that explicit ratings from 1 to 5 are rather abstract: there is no clear definition of when an item should be rated with a particular number, or what distinguishes a '3' from a '4', for example. Perhaps, interpretations of this scale vary between users in such a way, that the relatively rich information provided by explicit ratings, is contradictory or ambiguous enough to negatively impact the predictability of the data. After all, there is no 'gray area' when it comes to implicit ratings: a user either *has* or *has not* interacted with an item.

Nevertheless, the relative changes in HR and NDCG are not big enough in either algorithm to draw any definitive conclusions. So, although the results from Table 4 certainly provide an interesting opportunity for future research, it is only concluded from these results that the HR and NDCG scores of DeepICF(+a) are at least as high in data sets with implicit ratings as in data sets with explicit ratings.

In conclusion, answering RQ2 (Section 5.1.1), converting MovieLens ratings from explicit to implicit has no negative effect on the HR and NDCG attained by DeepICF(+a). The effect is possibly even slightly positive.

5.2 Pre-training effects

5.2.1 Motivation

Pre-training data has only been made available by Xue et al. (2019) for DeepICF(+a) on MovieLens, and pre-training DeepICF(+a) on the other data sets is impossible due to time and processing limitations. Thus, the application of these algorithms on MSD is performed without pre-training. However, if the lack of pre-training greatly reduces

the prediction accuracy delivered by DeepICF(+a), it is important to consider this negative influence when comparing these algorithm to FISM on MSD. Therefore, this section is concerned with investigating the effects of pre-training on DeepICF(+a). In doing so, RQ3 ("What effect does pre-training have on the HR and NDCG attained by DeepICF(+a) when applied to MovieLens?") is answered.

5.2.2 Methodology

DeepICF(+a) are both applied to MovieLens, with and without pre-training. MovieLens is used as data set, because this is the only data set with available pre-training data for these algorithms. For each run, the algorithms are set to terminate after 50 epochs; all other parameters assume their default values; and the accuracy of the algorithms is measured in HR and NDCG.

5.2.3 Results

Algorithm	Metric	MovieLens		
		with pre-training	*without pre-training*	*change*
DeepICF	HR	0.6846	0.6565	−4.1%
	NDCG	0.4089	0.3876	−5.2%
DeepICF+a	HR	0.6773	0.6199	−8.5%
	NDCG	0.4045	0.3537	−12.6%

Table 5: DeepICF(+a) HR and NDCG scores on MovieLens with and without pre-training

The results of applying DeepICF(+a) to MovieLens with and without pre-training are shown in Table 5. Furthermore, Table 5 contains the relative change in HR and NDCG when pre-training is disabled instead of enabled.

5.2.4 Discussion

It can be deduced from Table 5 that disabling pre-training has a negative effect on the accuracy of DeepICF(+a). HR and NDCG were reduced in both algorithms when pre-training was disabled. The accuracy of DeepICF+a dropped more than that of DeepICF: HR decreased by 8.5% and NDCG decreased by 12.6%, compared to a respective 4.1% and 5.2% decrease in HR and NDCG by DeepICF. Therefore, it seems that pre-training is especially useful for DeepICF+a, while it is also beneficial for DeepICF.

Moreover, for both algorithms, NDCG saw a larger score reduction than HR. This implies that pre-training is particularly beneficial for predicting the right order between relevant items.

Both of these observations are to be taken into account when applying DeepICF(+a) to MSD in Section 6, on which both algorithms have not been pre-trained, either. This analysis shows that, in all likelihood, the obtained results in that section are lower than they would be in practice, if pre-training is enabled.

Thus, answering and concluding RQ3 (Section 5.2.1): pre-training increases DeepICF(+a)'s HR and NDCG on MovieLens.

6 Application to the Million Song Dataset

6.1 Motivation

Now that the effects of reproduction and pre-training have been determined, inquiries can be made into the generalizability of DeepICF(+a). The investigated subjects in this section are captured by RQ4 ("How do the HR and NDCG achieved by DeepICF(+a) on MSD relate to the HR and NDCG achieved by DeepICF(+a) on MovieLens and Pinterest?") and RQ5 ("How do the HR and NDCG achieved by DeepICF(+a) on MSD relate to the HR and NDCG achieved by FISM on MSD?").

6.2 Methodology

DeepICF(+a) and FISM are applied to MSD. As pre-training is unavailable for this data set, it is deactivated in these runs. This deactivation is justified by the results from Section 5.2. The FISM code is slightly altered so that it does not raise an error when applied to MSD. For each run, the algorithms are set to terminate after 50 epochs; all other parameters assume their default values; and the accuracy of the algorithms is measured in HR and NDCG.

6.3 Results

Algorithm	MovieLens		Pinterest		MSD	
	HR	NDCG	HR	NDCG	HR	NDCG
DeepICF	0.6565	0.3876	0.8744	0.5526	0.1528	0.0698
DeepICF+a	0.6199	0.3537	0.8235	0.4970	0.1159	0.0473
FISM	0.6522	0.3841	0.8485	0.5196	0.1017	0.0423

Table 6: DeepICF(+a) and FISM HR and NDCG scores on MovieLens, Pinterest and MSD.

Table 6 presents the HR and NDCG scores of the applications of DeepICF(+a) and FISM to MSD. Results of the application of these algorithms to MovieLens and Pinterest have been included to improve the ease with which comparisons between the data sets can be made.

6.4 Discussion

Most prominently, Table 6 reflects that all three algorithms have a much lower prediction accuracy on MSD than on MovieLens or Pinterest. DeepICF showed the best performance on MSD, but the resulting HR and NDCG are still 75.4% and

80.3% lower, respectively, than the worst performance of all three algorithms on either MovieLens or Pinterest. This is expected, though, because 48.9% of all items in MSD only occur once (Table 2, Section 3). In contrast, just 3.1% of items in MovieLens merely have a single occurrence—and only 0.7% in Pinterest. The number of uniquely occurring items in MSD implies that almost half of all items in the test set does not occur in the train set. The algorithms cannot have computed these items' similarity to other items. This explains the relatively low HR and NDCG from DeepICF(+a) and FISM.

So, in an answer to RQ4 (Section 6.1): the HR and NDCG achieved by DeepICF(+a) on MSD are lower than on MovieLens and Pinterest.

In addition, the attention mechanism of DeepICF+a reduced HR by 24.1% relative to DeepICF. As inferred in Section 5.2, DeepICF+a sees a greater reduction in accuracy than DeepICF when pre-training is removed. Moreover, DeepICF+a's accuracy decreased more than DeepICF through the process of reproduction. For those reasons, this result is less surprising than it might seem at first glance.

As for the used FISM implementation, the accuracy metrics in Table 6 have the expected values of random recommendation lists. This is due to a modification of the code, without which the code raised an error. Therefore, unfortunately, FISM did not function as it should. Further confirmation of this is the fact that FISM showed no improvement through time, while DeepICF(+a) did. The error in FISM was likely the result of MSD's sparsity or number of uniquely occurring items (Table 6). Nevertheless, it is unknown what the actual cause is.

Thus, while the results shown in Table 6 imply DeepICF(+a)'s superiority over FISM—and especially that of DeepICF—this conclusion cannot definitively be made. In other words, in response to RQ5 (Section 6.1), the HR and NDCG achieved by DeepICF(+a) on MSD can neither be confirmed nor denied to be higher than the HR and NDCG achieved by FISM on the same data set.

7 General discussion

Still, this paper contributes to research surrounding recommender systems and DeepICF(+a) in particular. For example, throughout Sections 4, 5 and 6, DeepICF+a has yielded less accurate results than DeepICF. This signifies that DeepICF+a is more sensitive to pre-training and tweaking of parameters, even though Xue et al. (2019) claim DeepICF(+a) would be easier to implement than alternative algorithms.

DeepICF, however, showed fairly stable, accurate results. It is less influenced by tweaking of parameters, or at least, the default parameters were better suited in all experiments within this paper. Thus, overall, DeepICF is favorable to DeepICF+a.

Had this study had access to pre-training, or the opportunity to create pre-train data, the results could have been different—perhaps more 'flattering' for DeepICF+a. Yet, since this was not the case, it is probable that other practical implementations do not have the resources for this, either. Hence, the results are still very relevant to the practical possibilities of DeepICF(+a).

The FISM approach that was used in this paper, achieved similar results to the one used by Xue et al. Unfortunately, for uncertain reasons, this algorithm did not

function in combination with MSD. Nonetheless, because this approach is only compatible with binary ratings, the analysis in Section 5.1 may not have taken place if another FISM implementation was used. In that case, those surprising results—that providing binary ratings instead of 5-ary ratings could enhance DeepICF(+a)'s performance—would not have been obtained. In turn, a great opportunity for future research would not have been discovered: how could it be that less informative ratings contribute to more accurate predictions?

The relatively very low density of MSD compared to MovieLens and Pinterest also clearly reduced performance of all three algorithms. At first, a data set of reviews from Amazon in the category 'Music' was used,[10] but it had the same issues with extreme sparsity, and even *more* than half of all items only occurred once. In fact, MSD originally did not have this problem, which is part of the reason why it is the data set that was chosen (the other reason being the vagueness of online store categories). Before removing 99% of users, but after filtering by users with 20 or more interactions, just 0.1% (!) of songs were only interacted with once over all users. So, with more time and processing power, MSD would not have been reduced in size, and DeepICF(+a) would have presumably performed much better, accuracy-wise. This leaves room for future research to find out whether song preferences are actually harder to predict for DeepICF(+a) than movies and pictures. Furthermore, with play counts readily available in the original MSD, it would be interesting to investigate whether using those as implicit ratings can contribute to an improvement of accuracy in DeepICF(+a) or recommender systems in general.

8 Conclusion

In conclusion, DeepICF(+a) as performed by Xue et al. (2019) showed promising results, but merely on two different data sets: one of movies (MovieLens) and one of pictures (Pinterest). DeepICF(+a) prediction accuracy on a different domain, such as that of songs, was yet to be tested. To do so, multiple phases of research were completed by this paper. Firstly, the three data sets, MovieLens, Pinterest and MSD were pre-processed as necessary. Secondly, the negative effect of the reproduction process on the accuracy of DeepICF(+a), using default parameters, was confirmed. Thirdly, the slight positive influence–or, at least, the non-negative influence—of implicit ratings as opposed to explicit ratings on the performance of DeepICF(+a) was discovered. Fourthly, the relation between pre-training and increased accuracy by DeepICF(+a) was verified. Fifthly, DeepICF(+a) were shown to attain lower HR and NDCG scores on MSD relative to MovieLens and Pinterest. Lastly, it could not be concluded that DeepICF(+a) achieved higher accuracy on MSD than FISM, because of a malfunction in this particular implementation of FISM. Nevertheless, this paper has found many relevant results to DeepICF(+a); has laid the groundwork for further experimentation with DeepICF(+a); and has excited several opportunities for future research.

[10] This data set is available via `https://s3.amazonaws.com/amazon-reviews-pds/tsv/index.txt`.

References

Agarwal, S. (2006). *Learning from higher order relations* (Unpublished doctoral dissertation). UC San Diego.

Busa-Fekete, R., Szarvas, G., Elteto, T., & Kégl, B. (2012). An apple-to-apple comparison of learning-to-rank algorithms in terms of normalized discounted cumulative gain. In *Ecai 2012-20th european conference on artificial intelligence: Preference learning: Problems and applications in ai workshop* (Vol. 242).

Christakopoulou, E., & Karypis, G. (2014). Hoslim: Higher-order sparse linear method for top-n recommender systems. In *Pacific-asia conference on knowledge discovery and data mining* (pp. 38–49).

Dias, M. B., Locher, D., Li, M., El-Deredy, W., & Lisboa, P. J. (2008). The value of personalised recommender systems to e-business: a case study. In *Proceedings of the 2008 acm conference on recommender systems* (pp. 291–294).

Ekstrand, M. D., Riedl, J. T., & Konstan, J. A. (2011). *Collaborative filtering recommender systems.* Now Publishers Inc.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1) (No. 2). MIT press Cambridge.

Kabbur, S., Ning, X., & Karypis, G. (2013). Fism: factored item similarity models for top-n recommender systems. In *Proceedings of the 19th acm sigkdd international conference on knowledge discovery and data mining* (pp. 659–667).

Lü, L., Medo, M., Yeung, C. H., Zhang, Y.-C., Zhang, Z.-K., & Zhou, T. (2012). Recommender systems. *Physics reports*, *519*(1), 1–49.

Nanou, T., Lekakos, G., & Fouskas, K. (2010). The effects of recommendations' presentation on persuasion and satisfaction in a movie recommender system. *Multimedia systems*, *16*(4-5), 219–230.

Ning, X., & Karypis, G. (2011). Slim: Sparse linear methods for top-n recommender systems. In *2011 ieee 11th international conference on data mining* (pp. 497–506).

Nonlinear relationships and graphs without numbers. (2016). Retrieved from `https://open.lib.umn.edu/macroeconomics/chapter/nonlinear-relationships-and-graphs-without-numbers/`. University of Minnesota.

Pathak, B., Garfinkel, R., Gopal, R. D., Venkatesan, R., & Yin, F. (2010). Empirical analysis of the impact of recommender systems on sales. *Journal of Management Information Systems*, *27*(2), 159–188.

Pu, P., Chen, L., & Hu, R. (2011). A user-centric evaluation framework for recommender systems. In *Proceedings of the fifth acm conference on recommender systems* (pp. 157–164).

Ricci, F., Rokach, L., & Shapira, B. (2011). Introduction to recommender systems handbook. In *Recommender systems handbook* (pp. 1–35). Springer.

Rydning, D., Reinsel, J., & Gantz, J. (2018). The digitization of the world from edge to core. *Framingham: International Data Corporation*.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on world wide web* (pp. 285–295).

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in artificial intelligence*, *2009*.

Thakkar, P., Varma, K., Ukani, V., Mankad, S., & Tanwar, S. (2019). Combining user-based and item-based collaborative filtering using machine learning. In *Information and communication technology for intelligent systems* (pp. 173–180). Springer.

Wu, Y., DuBois, C., Zheng, A. X., & Ester, M. (2016). 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).

Xue, F., He, X., Wang, X., Xu, J., Liu, K., & Hong, R. (2019). Deep item-based collaborative filtering for top-n recommendation. *ACM Transactions on Information Systems (TOIS)*, *37*(3), 1–25.

Zhang, X.-D. (2020). Machine learning. In *A matrix algebra approach to artificial intelligence* (pp. 223–440). Springer.