Transfer Learning for Item Recommendations and Knowledge Graph Completion in Item Related Domains via a Co-Factorization Model

Guangyuan Piao(✉) and John G. Breslin

Insight Centre for Data Analytics, Data Science Institute,
National University of Ireland Galway, Galway, Ireland
guangyuan.piao@insight-centre.org, john.breslin@nuigalway.ie

Abstract. With the popularity of Knowledge Graphs (KGs) in recent years, there have been many studies that leverage the abundant background knowledge available in KGs for the task of item recommendations. However, little attention has been paid to the *incompleteness* of KGs when leveraging knowledge from them. In addition, previous studies have mainly focused on exploiting knowledge from a KG for item recommendations, and it is unclear whether we can exploit the knowledge in the other way, i.e, whether *user-item interaction histories* can be used for improving the performance of completing the KG with regard to the domain of items. In this paper, we investigate the effect of knowledge transfer between two tasks: (1) *item recommendations*, and (2) *KG completion*, via a co-factorization model (`CoFM`) which can be seen as a *transfer learning* model. We evaluate `CoFM` by comparing it to three competitive baseline methods for each task. Results indicate that considering the *incompleteness* of a KG outperforms a state-of-the-art factorization method leveraging existing knowledge from the KG, and performs better than other baselines. In addition, the results show that exploiting *user-item interaction histories* also improves the performance of completing the KG with regard to the domain of items, which has not been investigated before.

1 Introduction

Knowledge Graphs (KGs) such as DBpedia [16] and Wikidata [33] have received great attention in the past few years. KGs provide a great amount of knowledge which can be used in different applications such as inferring user interest profiles [15], personalization of digital health for individuals [32], and recommender systems [17,20]. For instance, DBpedia provides cross-domain background knowledge about entities/things, and the latest version of this KG describes 4.58 million things including 87,000 movies. Figure 1 shows pieces of information about the entity `dbr`[1]`:Bled_White_(2011_film)`, which can be retrieved from

[1] The prefix `dbr` denotes http://dbpedia.org/resource/.

A. Gangemi et al. (Eds.): ESWC 2018, LNCS 10843, pp. 496–511, 2018.
https://doi.org/10.1007/978-3-319-93417-4_32

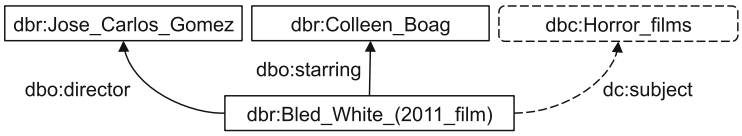


Fig. 1. Pieces of information about the movie `dbr:Bled_White_(2011_film)` from DBpedia. The piece of information with dotted lines denotes missing information from the knowledge graph.

DBpedia using SPARQL[2] queries through the SPARQL endpoint for DBpedia. Each piece of information about an entity in DBpedia is a RDF[3] triple, which consists of a *subject*, *predicate*, and *object* (e.g., `dbr:Bled_White_(2011_film)`, `dbo:starring`, `dbr:Colleen_Boag` in Fig. 1). With the freely accessible knowledge from KGs such as DBpedia, a big effort has been made in order to consume the knowledge for intelligent or adaptive systems such as recommender systems [3,21]. The focus of existing studies leveraging KGs for recommender systems has been on exploiting KG-enabled features for different types of machine learning or graph algorithms. Although previous studies have shown some useful insights about leveraging background knowledge about items from KGs for recommender systems, most of these studies have not considered the *incompleteness* of KGs.

Despite the fact that KGs provide billions of machine-readable facts about entities, they are far from complete [9], and a dedicated line of research has focused on the task of KG completion [6,8]. Indeed, most KGs use the *Open World Assumption*, i.e., it is not necessarily false if a KG does not contain a certain piece of information. The piece of information may be true but is missing from the KG. For example, the piece of information with dotted lines in Fig. 1 shows that the category `dbc`[4]`:Horror_films` is missing for the entity `dbr:Bled_White_(2011_film)` in DBpedia, which is important information in the context of recommending movies. Most previous studies that exploit KG-enabled features for item recommendations were based on the *existing* knowledge of KGs, and did not incorporate the *incompleteness* of KGs. In addition, these studies have focused on leveraging knowledge in one direction, i.e., from KGs to the task of item recommendations. Therefore, it is not clear that whether the knowledge from item recommendations, *user-item interaction histories*, can be transferred to the KG completion task with respect to the domain of items.

In this paper, we leverage a co-factorization model to investigate transfer learning [25] between these two tasks: (1) *item recommendations*, and (2) *KG completion* with respect to the domain of items. Here, *transfer learning* denotes using one task as a "source" task and the other as a "target" task. First, with item recommendations as the target task and KG completion as the source task, we are interested in whether incorporating the *incompleteness* of a KG

[2] https://www.w3.org/TR/rdf-sparql-query/.
[3] https://www.w3.org/RDF/.
[4] The prefix `dbc` denotes http://dbpedia.org/resource/Category.

performs better when compared to a state-of-the-art approach using a Factorization Machine (FM) which exploits existing knowledge from the KG, and outperforms other baselines. Second, we aim to investigate whether the knowledge can be transferred from item recommendations to the KG completion and improves the performance when KG completion is the target task. We use DBpedia as our KG in this study, which has been widely used for KG-enabled recommender systems. In summary, our contributions are as follows:

- We study knowledge transfer between two tasks: (1) *item recommendations*, and (2) *KG completion* for the specific domain of items, via a co-factorization model (CoFM). This transfer learning model incorporates the incompleteness of a KG for item recommendations, and incorporates the knowledge from item recommendations for completing the KG (Sect. 3).
- In Sect. 5, we evaluate CoFM with three baselines for each task, and show that incorporating the *incompleteness* of KG outperforms the baselines significantly. In addition, we show that exploiting the knowledge from item recommendations improves the performance of KG completion with respect to the domain of items, which has not been studied in previous studies.

2 Related Work

In this section, we review some related work on (1) exploiting KGs for item recommendations, and (2) KG completion.

Exploiting Knowledge Graph for Item Recommendations. With the popularity of Linked Open Data (LOD), much research work has been done in order to leverage the background knowledge from LOD (mainly from open KGs such as DBpedia) for items with different purposes with respect to recommender systems, e.g., for improving recommendation performance [4,13,17,18,20,21,24,28,35], for explaining recommendations [19], and for cross-domain recommendations [12]. Various approaches have been investigated for LOD-enabled Recommender Systems (LODRecSys) with aims at improving the recommendation performance, such as semantic similarity/distance measures for measuring the similarity/distance of two entities in the same domain [26,27], applying graph-based algorithms by incorporating background knowledge of items from KGs [18,20,21], and feeding KG-enabled features into different types of machine learning approaches [4,24,28]. A detailed review on LODRecSys can be found in [7,10].

More recently, Zhang et al. [35] used the sum of several embeddings of an item with respect to *structural*, *textual*, and *visual* knowledge from a KG in addition to the item embedding learned from user-item interactions for item recommendations. However, the loss functions for item recommendations and KG completion have the same weight which might not be optimal for the target task, and the knowledge transfer from item recommendations to the KG completion task was not investigated. Piao et al. [28] proposed using a FM with lightweight LOD features which can be directly obtained from the DBpedia SPARQL endpoint such

as the *predicate-object lists* and *PageRank scores* of items. In [28], the authors also showed that their approach outperforms baseline methods such as BPRMF [30] and a state-of-the-art approach, which uses *learning-to-rank* algorithms for consuming KG-enabled features [24]. We use the FM approach of [28], which consumes the *existing* knowledge from a KG without considering the incompleteness of the KG, as one of our baselines.

Knowledge Graph Completion. Although KGs provide large amounts of facts about entities/things, they are highly incomplete [9]. For instance, Google observed that 71% of people in Freebase [1] lack a place of birth, and 75% lack a nationality [5]. To tackle the problem, two groups of embedding-based approaches have been proposed. One is using factorization approaches such as tensor factorization [2,6,22,23], and the other is using neural network models [11,14,34]. A typical work is TransE [2], which is a translation-based model for learning embeddings of entities and the relationships between them. The main idea of TransE is based on the assumption that if (s, p, o) is a valid triple, then the sum of embeddings for s and p should be close to the embedding of o. Drumond et al. [6] showed that using PITF (Pairwise Interaction Tensor Factorization) [31] is highly successful in predicting incomplete triples in KGs compared to other *canonical decomposition* approaches such as the one from [8].

Our work differs from previous work on KG completion as we focus on whether the knowledge transferred from *user-item interaction histories* in the task of item recommendations improves the KG completion performance with respect to the specific domain of items.

3 Learning with a Co-Factorization Model

In this section, we first formulate the two tasks - (1) item recommendations, and (2) KG completion, and then describe state-of-the-art approaches for each task in Sects. 3.1 and 3.2, respectively. Finally, we present a co-factorization model (CoFM) for transfer learning between these two tasks in Sect. 3.3.

- *Item recommendations* (task#1): Given user-item interaction histories, i.e., likes or dislikes about items (we consider binary interactions in this study), our goal is to provide the top-N item recommendations for a target user.
- *KG completion* (task#2): This task can be formulated into a top-N recommendations task as well, in the same way as previous studies [6,14]. For a given (*subject*, *predicate*) pair, the task is providing the top-N *object* recommendations from a set of candidate *objects*. Candidate *objects* are all objects in the *range* of a given *predicate* defined in the DBpedia ontology.

3.1 Factorization Machines for Item Recommendations

We use FMs for item recommendations. It is a state-of-the-art framework for latent factor models, and has been widely used for collaborative filtering tasks. The FM model of order $d = 2$ is defined as:

$$\hat{y}^{FM}(x) = w_0 + \sum_{i=1}^{q} w_i x_i + \sum_{i=1}^{q} \sum_{j>i}^{q} <v_i, v_j> x_i x_j \tag{1}$$

where $\hat{y}^{FM}(x)$ denotes the predicted score using `FMs`, w_0 denotes the weight for a bias term, x represents feature values, and v represents the latent factors of each feature. In addition, $w_0 \in \mathbb{R}, x$ and $w \in \mathbb{R}^q, v_i \in \mathbb{R}^m$, m is the dimensionality of the factorization, and q is the number of features. The first part of the `FM` model is similar to a simple linear regression, which captures the interactions of each input variable x_i. In addition, the second part of the model captures all pairwise interactions of input variables $x_i x_j$ using the latent factors of features. As a general framework, `FMs` can mimic many state-of-the-art latent factor models such as `BPRMF` [30] and `PITF` [31], which has been shown in a previous study [29]. For a more detailed description of `FMs`, please refer to Rendle [29].

The first task is to provide the top-N item recommendations based on the history of user-item interactions. We focus on a binary response $y_{d_{ui}}$ (e.g., a user u likes or dislikes an item i) of each item in this study. Let β_0 denote the bias, $\boldsymbol{\beta}_i$ denote the weights of features with respect to i, and $\boldsymbol{\theta}_i$ denote a list of latent factors for i, which can be learned through `FMs` with the training dataset. In addition, $\mathbf{x_{d_{ui}}}$ denotes a list of explicit features in a training example d_{ui}. The simplest case for $\mathbf{x_{d_{ui}}}$ is that it consists of one categorical feature to denote a user u, and the other categorical feature to denote an item i. Following the definition of the `FM` model (Eq. 1), we can estimate the preference score of an item i based on $\mathbf{x_{d_{ui}}}$, $\boldsymbol{\beta}$ and $\boldsymbol{\Theta}$ as $f(s_{d_{ui}}|\mathbf{x_{d_{ui}}}, \boldsymbol{\beta}, \boldsymbol{\Theta})$, where

$$s_{d_{ui}} = \beta_0 + \boldsymbol{\beta}_u + \boldsymbol{\beta}_i + <\boldsymbol{\theta}_u, \boldsymbol{\theta}_i> \tag{2}$$

The task of recommending the top-N items can be formalized as optimizing the Bayesian Personalized Ranking (`BPR`) [30] loss as follows:

$$\ell(a_1, a_2) = \sum_{a_1 \in \mathcal{D}^+_{ui}} \sum_{a_2 \in \mathcal{D}^-_{ui}} -\log[\delta(s_{a_1} - s_{a_2})] \tag{3}$$

where δ is a sigmoid function: $\delta(x) = \frac{1}{1+e^{-x}}$, and $\mathcal{D}^+_{ui}$ and $\mathcal{D}^-_{ui}$ denote the set of positive and negative training instances, respectively. In detail, a positive training instance consists of a user and an item which the user liked in the training dataset. On the other hand, a negative instance for the user consists of the user and a randomly chosen item which is not in the list of items the user liked before in the training set. The intuition behind `BPR` is that a liked item for a user should be ranked higher (with a higher score) compared to a random one in the list of items with which the user has not interacted. In fact, the `FM` using `BPR` for optimization with users and items as features, is exactly a biased `BPRMF` [30], which has been shown in the previous study [29].

3.2 Translating Embeddings for KG Completion

The second task is the KG completion with respect to the domain of items, which can be formulated as *object* recommendations given a *subject* and *predicate* pair

[6]. We use a translation-based embedding model, `TransE` [2], for the second task. `TransE` is one of the most popular approaches for KG completion due to its effectiveness despite its simplicity. The intuition behind this model is to learn latent factors for *subjects*, *predicates*, and *objects*, in order to satisfy $\boldsymbol{\phi}_s + \boldsymbol{\phi}_p \approx \boldsymbol{\phi}_o$ when (s, p, o) is a valid triple in the KG. In other words, for a valid triple (s, p, o), we want to make the embedding of o ($\boldsymbol{\phi}_o$) be the nearest neighbor of $\boldsymbol{\phi}_s + \boldsymbol{\phi}_p$ where the distance is measured by a dissimilarity function $d(\boldsymbol{\phi}_s + \boldsymbol{\phi}_p, \boldsymbol{\phi}_o)$ such as L_2-norm. Therefore, the prediction score of a candidate o for given (s, p) can be measured as follows when L_2-norm is used as the dissimilarity function:

$$s'_{d_{spo}} = \sqrt{\sum_{j=1}^{m} (\phi_{s_j} + \phi_{p_j} - \phi_{o_j})^2} \tag{4}$$

where m denotes the dimensionality of the factorization/embedding for s, p, and o. Afterwards, the candidate set of *objects* can be ranked by their prediction scores, where an *object* with a higher distance score should be ranked lower. The loss to be optimized in `TransE` can be defined as below in our settings [2]:

$$\ell(b_1, b_2) = \sum_{b_1 \in \mathcal{D}^+_{spo}} \sum_{b_2 \in \mathcal{D}^-_{spo}} [\gamma + s'_{b_1} - s'_{b_2}]_+ \tag{5}$$

where $\mathcal{D}^+_{spo}$ and $\mathcal{D}^-_{spo}$ denote the set of positive and negative training instances, respectively. Here, a positive instance denotes a valid triple (s, p, o), which can be found in the training set, and a negative instance consists of s, p, and a randomly chosen *object* o^- which does not exist in the training set. $[x]_+ = 0 \quad for \quad x < 0$, and x otherwise, and γ is a margin hyperparameter. In the same way as [2], we set γ to 1.0, and use L_2-norm as our dissimilarity function.

3.3 Transfer Learning via a Co-Factorization Model for the Two Tasks

As we can see from Eqs. 2 and 4, we have two related representations for the latent factors of an item i in task#1 (or *subject* s in task#2), i.e., $\boldsymbol{\theta}_i$ and $\boldsymbol{\phi}_s$, in the context of the two different tasks. In this work, we investigate two strategies for modeling the relationship between the two representations of items/subjects for transfer learning between the two tasks. For the sake of simplicity, we consider simple cases of $\mathbf{x_{d_{ui}}}$, i.e., two categorical features (to denote u and i) for $\mathbf{x_{d_{ui}}}$.

Shared Latent Space (`CoFM`$_\mathtt{A}$). A straightforward approach to model the relationship between two representations for the latent factors of an item/subject in the two tasks is to assume that their latent factors are exactly the same, i.e., $\boldsymbol{\theta}_i = \boldsymbol{\phi}_s = \boldsymbol{\rho}_{is}$, where $\boldsymbol{\rho}_{is}$ is the same latent factor for both. Given this assumption, the preference score functions (Eqs. 2 and 4) for the aforementioned two tasks can then be re-written as:

$$s_{d_{ui}} = \beta_0 + \boldsymbol{\beta}_u + \boldsymbol{\beta}_i + <\boldsymbol{\theta}_u, \underline{\boldsymbol{\rho}_{is}}>, \qquad s'_{d_{spo}} = \sqrt{\sum_{j=1}^{m} (\underline{\rho_{is_j}} + \phi_{p_j} - \phi_{o_j})^2} \tag{6}$$

This approach is based on a strong assumption that an item and a subject from the two different tasks have the same latent representation.

Via Latent Space Regularization (CoFM$_R$). An alternative approach to work with the two latent representations of an item/subject is regularizing these representations to make them not reside too far away from each other. We incorporate this intuition into the model by imposing the following regularization:

$$\lambda_{\phi,\theta} \|\phi_s - \theta_i\|_F^2 \tag{7}$$

where $\lambda_{\phi,\theta}$ is a regularization parameter.

Another issue for transfer learning between the two tasks is the different output scales of the two loss functions: Eqs. 3 and 5. Hence, we modify the loss function of the KG completion task (Eq. 5) as follows in order to make both loss functions in the two tasks have the same scale.

$$\ell(b_1, b_2) = \sum_{b_1 \in \mathcal{D}^+_{spo}} \sum_{b_2 \in \mathcal{D}^-_{spo}} -\log[\delta(\gamma + s'_{b_1} - s'_{b_2})]_+ \tag{8}$$

Summary. Putting everything together, our co-factorization model in the view of *transfer learning* can be formulated as follows:

$$Opt(CoFM) : Opt(T) + \epsilon \times Opt(S), \tag{9}$$
$$Opt(T) = \arg \min \sum_{d_T \in \mathcal{D}_T} \ell_T(\cdot), \qquad Opt(S) = \arg \min \sum_{d'_S \in \mathcal{D}'_S} \ell_S(\cdot)$$

where ϵ is a transfer (auxiliary) parameter to denote the importance of the knowledge transfer from the source task (S) to the target task (T). Let $\mathcal{D}_T$ and $\mathcal{D}_S$ denote the original training instances in the target and source tasks, respectively. $\mathcal{D}'_S$ is a set of training instances that is randomly sampled from $\mathcal{D}_S$ in order to match the size of $\mathcal{D}_T$, i.e., $|\mathcal{D}_T| = |\mathcal{D}'_S|$. For each instance $d_T \in \mathcal{D}_T$, we choose an instance d'_S randomly with a replacement from $\mathcal{D}_S$ where the *item* in d_T is the same as the *subject* in d'_S, i.e., $d_T(i) = d'_S(s)$. With the same size for both T and S, we then use the well-known Stochastic Gradient Decent (SGD) to learn the parameters in the CoFM. An overview of the algorithm to optimize Eq. 9 using SGD is presented in Algorithm 1 when the target task is item recommendations. Our approach can be seen as a *transfer learning* [25] model as we are transferring knowledge between two different but related tasks in the same domain. It is worth noting that, in contrast to *multi-task learning* which aims to learn both tasks simultaneously, *transfer learning* aims to achieve the best performance for T with the transferred knowledge from S.

4 Experiment Setup

Here we describe our evaluation metrics (Sect. 4.1), the datasets for our experiment (Sect. 4.2), and the methods we compared for evaluating our approach (Sect. 4.3).

Algorithm 1. Main elements of the algorithm to optimize Eq. 9 using SGD when the target task T is item recommendations.

input : training datasets $\mathcal{D}_{ui}$ and $\mathcal{D}'_{spo}$ with the same size of $|\mathcal{D}|$, initialized parameters for CoFM
output: learned parameters for CoFM
repeat
  for d_{ui} *in* $\mathcal{D}_{ui}$ **do**
    Optimize Opt(T) for $\boldsymbol{\theta_u}, \boldsymbol{\theta_i}, \boldsymbol{\beta}$
      perform SGD for BPR loss function in terms of d_{ui}
    Select d'_{spo} **in** $\mathcal{D}'_{spo}$ **where** $d'_{spo}(s) = d_{ui}(i)$
    Optimize Opt(S) for $\boldsymbol{\phi_s}, \boldsymbol{\phi_p}, \boldsymbol{\phi_o}$
      perform SGD for BPR loss function in terms of d'_{spo}
until *converged*;

4.1 Evaluation Metrics

The recommendation performance was evaluated via five evaluation metrics that have been widely used in previous studies as below. We describe these metrics in the context of item recommendations for users.

- **MRR**: MRR (Mean Reciprocal Rank) is the average of the reciprocal ranks of relevant items for users. It can be measured as: $MRR = \frac{1}{|U|}\sum_{k=1}^{|U|}\frac{1}{rank_k}$, where U denotes the set of users, and $rank_k$ refers to the rank position where the first *relevant* item with respect to a user $u \in U$ occurs.
- **MAP**: MAP (Mean Average Precision) measures the mean of the average precision scores (AP) of liked items of all users. The average precision of a user u is defined as: $AP_u = \frac{\sum_{n=1}^{N} P@n \times like(n)}{|I|}$, where n is the number of items, $|I|$ is the number of liked items of u, and $like(n)$ is a binary function to indicate whether the user prefers the n-th item or not.
- **P@N**: $P@N = \frac{|\{relevant\ items@N\}|}{N}$ (Precision at rank N) is the proportion of the top-N recommendations that are relevant to the user.
- **R@N**: $R@N = \frac{|\{relevant\ items@N\}|}{|\{relevant\ items\}|}$ (Recall at rank N) represents the mean probability that relevant items are successfully retrieved within the top-N recommendations.
- **nDCG@N**: nDCG (Normalized Discounted Cumulative Gain) takes into account rank positions of the relevant items, and can be computed as follows: $nDCG@N = \frac{1}{IDCG@N}\sum_{k=1}^{N}\frac{2^{\hat{r}_{uk}}-1}{\log_2(k+1)}$ where $\hat{r}_{uk}$ is the relevance score of the item at position k with respect to a user u in the top-N recommendations, and IDCG@N denotes the score obtained by an ideal top-N ranking.

In the same way as [28], we use the *bootstrapped paired t-test* for testing the significance. The significance level of α is set to 0.05 unless otherwise noted.

4.2 Dataset

We use two datasets in the movie and book domains, which have been widely used in LODRecSys [18,24,28].

- **Movielens dataset** [24]. The dataset is a refined Movielens dataset[5] for LODRecSys. It consists of users and their ratings about movies, and each of the items in this dataset has been mapped into DBpedia entities if there is a mapping available[6]. In the same way as previous studies [24,28], we consider ratings higher than 3 as positive feedback and others as negative ones.
- **DBbook dataset**. The dataset[7] consists of users and their binary feedback (1 for likes, and 0 otherwise), where the items have been mapped to DBpedia entities if there is a mapping available.

Table 1 shows the main details of user-item interactions and RDF triples in the two datasets. There are 3,997 users and 3,082 items with 827,042 ratings in the Movielens dataset. The DBbook dataset consists of 6,181 users and 6,733 items with 72,372 interactions. The sparsity of the DBbook dataset (99.38%) is higher than that of the Movielens dataset (93.27%). For item recommendations, we use 80% and 20% of each dataset for training and test sets where 20% of the training set was used for validation. In addition, all of the items were considered as candidate items for recommendations in the same way as [24] instead of considering only "rated test-one" evaluation. The second part of Table 1 shows the details of extracted triples for items/subjects in the two datasets from the DBpedia SPARQL endpoint. In the Movielens dataset, 2,952 out of 3,082 (95.8%) items have at least one triple. There are 21 distinct predicates and 18,550 objects in the Movielens dataset, which results in 81,835 triples in total. In the case of DBbook dataset, 6,211 out of 6,733 (92.2%) items have at least one triple. There are 36 distinct predicates and 16,476 objects in the DBbook dataset, which results in 72,911 triples in total. For KG completion with respect to the domain of items, we adopt the same splitting strategy as [6] for constructing training and test sets. We randomly choose a *subject* and *predicate* pair (s, p) for a given s, and then use all triples containing the pair to construct the test set. The other triples with the same *subject* were put into the training set.

We repeated five times by sampling new training and test sets for the two tasks using the aforementioned strategies, and applied different methods to them. The results in Sect. 5 are based on the averages over five runs.

4.3 Compared Methods

We use $\texttt{CoFM}_{\texttt{A}}$ to denote the `CoFM` which shares latent space with the assumption that two latent factors of an item/subject in the two tasks are exactly the same,

[5] https://grouplens.org/datasets/movielens/1m/.
[6] http://sisinflab.poliba.it/semanticweb/lod/recsys/datasets/.
[7] http://challenges.2014.eswc-conferences.org.

Table 1. Statistics of Movielens and DBbook datasets.

		Movielens	DBbook
Statistics of user-item interactions	# of users	3,997	6,181
	# of items	3,082	6,733
	# of ratings	827,042	72,372
	Avg. # of ratings	206	12
	Sparsity	93.27%	99.38%
	% of positive ratings	56%	45.85%
Statistics of RDF triples	# of subjects	2,952 (3,082)	6,211 (6,733)
	# of predicates	21	36
	# of objects	18,550	16,476
	# of triples	81,835	72,911

and use $\texttt{CoFM}_\texttt{R}$ to denote the CoFM which uses regularization for modeling the relationship between the two latent factors of an item/subject in the two tasks.

Parameter Settings of CoFM. The transfer (auxiliary) parameter ϵ was determined by a separate validation set randomly retrieved from 20% of the training set for the first run in terms of the loss on the target task in each dataset. According to the results, ϵ was set to 0.05 for the Movielens dataset when either KG completion or item recommendations is the target task. For the DBbook dataset, ϵ was set to 0.05 and 1.0 when KG completion and item recommendations is the target task, respectively. In addition, we set the same value for all regularization parameters in our approach for the sake of simplicity. $\lambda = 0.01$ when item recommendations is the target task, and $\lambda = 0.001$ when KG completion is the target one. The dimensionally value m was set to 64, which is the same as in [6], for all factorization-based approaches.

We compare CoFM against the following methods for item recommendations.

- kNN-item (kNN): kNN-item is an item-based k-nearest neighbors algorithm. We use a MyMedialiite[8] implementation for this baseline where $k = 80$.
- BPRMF [30]: BPRMF is a matrix factorization approach for learning latent factors with respect to users and items, optimized for BPR. BPRMF can be seen as the model for item recommendations in CoFM, which is a FM model without transferring knowledge from the KG completion task.
- $\texttt{FM}_\texttt{LOD}$ [28]: $\texttt{FM}_\texttt{LOD}$ exploits *lightweight* KG-enabled features about items from DBpedia, which can be obtained directly from its SPARQL endpoint.

[8] http://www.mymedialite.net/.

For the KG completion task, we compare CoFM against the following methods:

- MFPP: Most Frequent Per Predicate (MFPP) is a baseline method which recommends the *objects* that co-occur most frequently with the *predicate* p given a *subject* s and *predicate* pair (s, p).
- PITF [31]: This model has been proposed in [31] for tag recommendations. In [6], the authors applied a PITF model optimized for the BPR criterion, which captures the interactions among *subjects*, *predicates*, and *objects* of RDF triples. We re-implement this approach under the framework of FMs.
- TransE [2]: This is a *translation-based* approach which models relationships by interpreting them as translations operating on the entity embeddings. We re-implemented this approach based on the parameters from [2]. As one might expect, TransE can be seen as the model for the KG completion task in CoFM without transferring knowledge from item recommendations.

We use SGD to learn the parameters in the aforementioned factorization models.

5 Results

Table 2 shows the results of comparing CoFM with the aforementioned methods in each task on the Movielens and DBbook datasets. Overall, CoFM provides the best performance compared to other approaches in item recommendations as well as KG completion in both datasets. As we can see from Table 2(a), CoFM_R provides the best performance, and improves the recommendation performance significantly ($\alpha < 0.01$) compared to kNN and BPRMF for item recommendations on the Movielens dataset. Similarly, CoFM_R outperforms baselines such as MFPP and PITF significantly for KG completion. In detail, a significant improvement of CoFM_R over PITF in MRR (+21%), MAP (+15%), nDCG@5 (+19.8%), P@5 (+31.2%), and R@5 (+8.2%) can be noticed. On the DBbook dataset (Table 2(b)), CoFM_A provides the best performance instead of CoFM_R. CoFM_A outperforms kNN and BPRMF significantly for item recommendations, and outperforms MFPP and PITF for the KG completion task ($\alpha < 0.01$). One of the possible explanations for the observation that the best performance is achieved by CoFM_R for the Movielens dataset and by CoFM_A for the DBbook one might be due to the different sparsity levels of the two datasets. As we can see from Table 1, the DBbook dataset has higher sparsities compared to the Movielens dataset for both tasks. CoFM_A, which can be seen as having strong knowledge transfer with the assumption that item/subject embeddings in the two tasks are the same, may possibly be more useful for this sparse dataset and leads to better performance.

FM_LOD vs. CoFM. We observe that CoFM_R, which incorporates the *incompleteness* of DBpedia, outperforms FM_LOD which leverages *existing* knowledge from DBpedia on the Movielens dataset. A significant difference between the two approaches in terms of all evaluation metrics can be noticed ($\alpha < 0.01$). On the DBbook dataset, CoFM_A also consistently outperforms FM_LOD in terms of all evaluation metrics specifically in terms of precision, e.g., +8.3% of P@5, +8% of P@10, and +5.6% of P@20 ($\alpha < 0.01$). The results show that incorporating the incompleteness of the KG improves the performance of item recommendations significantly.

Table 2. Results of *KG completion* and *item recommendations* on the Movielens (a) and DBbook (b) datasets. S denotes source task while T denotes target task. The gray cells denote significant improvement over the best-performing baseline.

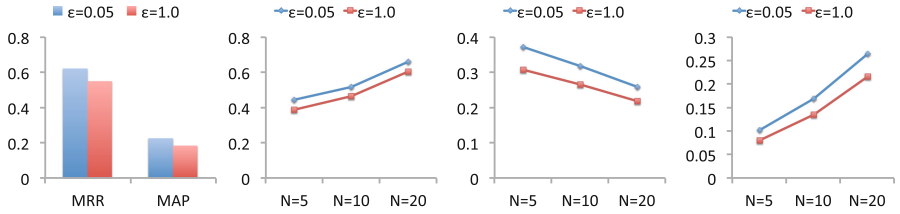
(a) Movielens

	***S*: KG completion *T*: item recommendations**					***S*: item recommendations *T*: KG completion**				
	kNN	BPRMF	FM_{LOD}	CoFM_{A}	CoFM_{R}	MFPP	PITF	TransE	CoFM_{A}	CoFM_{R}
MRR	0.510	0.594	0.609	0.602	**0.622**	0.183	0.266	0.317	0.302	**0.322**
MAP	0.181	0.218	0.224	0.220	**0.227**	0.090	0.193	0.217	0.208	**0.222**
nDCG@5	0.358	0.425	0.436	0.429	**0.445**	0.149	0.248	0.292	0.279	**0.297**
P@5	0.291	0.355	0.366	0.360	**0.372**	0.070	0.096	0.123	**0.126**	**0.126**
R@5	0.075	0.097	0.100	0.098	**0.102**	0.103	0.230	0.241	0.240	**0.249**
nDCG@10	0.440	0.500	0.510	0.504	**0.518**	0.171	0.273	0.311	0.299	**0.316**
P@10	0.258	0.307	0.314	0.310	**0.318**	0.046	0.064	0.077	**0.081**	0.079
R@10	0.129	0.161	0.165	0.164	**0.170**	0.149	0.271	0.277	0.280	**0.283**
nDCG@20	0.583	0.645	0.653	0.648	**0.660**	0.194	0.297	0.331	0.321	**0.336**
P@20	0.218	0.252	0.257	0.254	**0.259**	0.031	0.042	0.047	**0.051**	0.048
R@20	0.213	0.257	0.261	0.260	**0.265**	0.199	0.311	0.313	0.318	**0.319**

(b) DBbook

	***S*: KG completion *T*: item recommendations**					***S*: item recommendations *T*: KG completion**				
	kNN	BPRMF	FM_{LOD}	CoFM_{A}	CoFM_{R}	MFPP	PITF	TransE	CoFM_{A}	CoFM_{R}
MRR	0.015	0.115	0.121	**0.125**	0.100	0.168	0.383	0.408	**0.412**	**0.412**
MAP	0.011	0.077	0.081	**0.083**	0.065	0.152	0.321	0.340	**0.345**	0.343
nDCG@5	0.008	0.105	0.110	**0.114**	0.091	0.162	0.372	0.399	**0.410**	0.400
P@5	0.003	0.034	0.036	**0.039**	0.031	0.048	0.111	0.117	**0.119**	**0.119**
R@5	0.010	0.096	0.101	**0.106**	0.085	0.177	0.363	0.377	0.380	**0.381**
nDCG@10	0.014	0.125	0.131	**0.134**	0.108	0.181	0.389	0.416	**0.423**	0.416
P@10	0.004	0.024	0.025	**0.027**	0.022	0.031	0.064	0.067	**0.068**	0.067
R@10	0.023	0.135	0.141	**0.147**	0.116	0.220	0.396	0.406	**0.408**	**0.408**
nDCG@20	0.022	0.145	0.153	**0.156**	0.126	0.203	0.404	0.428	**0.434**	0.430
P@20	0.004	0.017	0.018	**0.019**	0.015	0.021	0.037	0.037	**0.038**	**0.038**
R@20	0.043	0.187	0.196	**0.198**	0.138	0.279	0.428	0.433	0.435	**0.436**

With vs. Without Knowledge Transfer. We now look at the results of CoFM with and without transferring knowledge between the two tasks. BPRMF and TransE can be seen as the CoFM without transferring knowledge between these tasks. On the Movielens dataset, CoFM_{R} improves the performance by 2.3%–5.2% compared to BPRMF for item recommendations ($\alpha < 0.01$). Regarding the KG completion task, CoFM_{R} outperforms TransE significantly for all evaluation metrics as well. On the DBbook dataset. CoFM_{A} improves the performance by 5.9%-14.7% compared to BPRMF for item recommendations. For the KG completion task, CoFM_{A} outperforms TransE significantly in terms of all evaluation metrics



(a) MAP and MRR (b) nDCG@N (c) P@N (d) R@N

Fig. 2. The performance of item recommendations on the Movielens dataset with $\epsilon = 0.05$ and $\epsilon = 1.0$ using $\texttt{CoFM}_{\texttt{R}}$.

except R@10. This indicates that transferring knowledge between the two tasks improves the performance on both tasks compared to each single model without transferring knowledge from the other task.

With vs. Without Tuning the Transfer Parameter ϵ. Figure 2 shows the results of item recommendations on the Movielens dataset using $\texttt{CoFM}_{\texttt{R}}$ with a tuned value for the parameter ϵ ($\epsilon = 0.05$) and without tuning the parameter ($\epsilon = 1.0$). As we can see from the figure, tuning the transfer value ϵ plays an important role in achieving the best performance for the target task.

To sum up, the implications of these results are twofold. With a proper transfer parameter, (1) incorporating the *incompleteness* of a KG can improve the performance of item recommendations, and (2) the knowledge from item recommendations, i.e., *user-item interaction histories* can also be transferred to the task of KG completion with respect to the domain of items, which improves the performance significantly.

6 Conclusions

In this paper, we investigated the effect of transferring knowledge between two tasks: (1) *item recommendations*, and (2) *knowledge graph completion* with respect to the domain of items. Compared to previous approaches, which directly leverage the existing background knowledge in a KG, the transfer learning model `CoFM` incorporates the incompleteness of a KG into the design of `CoFM`. The experimental results are promising and suggest that incorporating the incompleteness of a KG improves the recommendation performance significantly compared to a state-of-the-art `FM` approach, which uses existing knowledge from a KG, and outperforms other baselines as well. In addition, we further explored potential synergies that transfer knowledge from item recommendations, i.e., *user-item interaction histories* to the task of KG completion, which has not been explored in previous studies. Results indicate that the knowledge from *user-item interaction histories* can be transferred to the KG completion task, and improves its performance significantly. As a further step, we plan to investigate other ways to model the relationship between two representations of an item/subject in the two tasks, e.g., using different dimensions for representing items.

Acknowledgments. This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289 (Insight Centre for Data Analytics).

References

1. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1247–1250. ACM (2008)
2. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data, pp. 2787–2795 (2013)
3. Di Noia, T., Cantador, I., Ostuni, V.C.: Linked open data-enabled recommender systems: ESWC 2014 challenge on book recommendation. In: Presutti, V., Stankovic, M., Cambria, E., Cantador, I., Di Iorio, A., Di Noia, T., Lange, C., Reforgiato Recupero, D., Tordai, A. (eds.) SemWebEval 2014. CCIS, vol. 475, pp. 129–143. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12024-9_17
4. Di Noia, T., Mirizzi, R., Ostuni, V.C., Romito, D.: Exploiting the web of data in model-based recommender systems. In: Proceedings of the 6th ACM Conference on Recommender Systems, pp. 253–256. ACM (2012)
5. Dong, X.L., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., Zhang, W.: Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In: The 20th SIGKDD, pp. 601–610. ACM (2014)
6. Drumond, L., Rendle, S., Schmidt-Thieme, L.: Predicting RDF triples in incomplete knowledge bases with tensor factorization. In: Proceedings of the 27th Annual ACM Symposium on Applied Computing, pp. 326–331. ACM (2012)
7. Figueroa, C., Vagliano, I., Rodríguez Rocha, O., Morisio, M.: A systematic literature review of linked data-based recommender systems. Concurrency Computation **27**, 4659–4684 (2015)
8. Franz, T., Schultz, A., Sizov, S., Staab, S.: TripleRank: ranking semantic web data by tensor decomposition. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 213–228. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04930-9_14
9. Galárraga, L., Razniewski, S., Amarilli, A., Suchanek, F.M.: Predicting completeness in knowledge bases. In: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, pp. 375–383. ACM (2017)
10. de Gemmis, M., Lops, P., Musto, C., Narducci, F., Semeraro, G.: Semantics-aware content-based recommender systems. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 119–159. Springer, Boston (2015). https://doi.org/10.1007/978-1-4899-7637-6_4
11. Guo, S., Wang, Q., Wang, B., Wang, L., Guo, L.: Semantically smooth knowledge graph embedding. In: ACL, vol. 1, pp. 84–94 (2015)
12. Heitmann, B.: An open framework for multi-source, cross-domain personalisation with semantic interest graphs. In: Proceedings of the Sixth ACM Conference on Recommender systems, pp. 313–316. ACM (2012)
13. Heitmann, B., Hayes, C.: Using linked data to build open, collaborative recommender systems. In: AAAI Spring Symposium, pp. 76–81. AAAI (2010)
14. Ji, G., Liu, K., He, S., Zhao, J.: Knowledge graph completion with adaptive sparse transfer matrix. In: Thirtieth AAAI Conference on Artificial Intelligence (2016)

15. Kapanipathi, P., Jain, P., Venkataramani, C., Sheth, A.: User interests identification on Twitter using a hierarchical knowledge base. In: Presutti, V., d'Amato, C., Gandon, F., d'Aquin, M., Staab, S., Tordai, A. (eds.) ESWC 2014. LNCS, vol. 8465, pp. 99–113. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07443-6_8
16. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S.: DBpedia-a Large-scale, multilingual knowledge base extracted from Wikipedia. Semant. Web J. **1**, 1–5 (2013)
17. Lu, C., Stankovic, M., Radulovic, F., Laublet, P.: Crowdsourced affinity: a matter of fact or experience. In: Blomqvist, E., Maynard, D., Gangemi, A., Hoekstra, R., Hitzler, P., Hartig, O. (eds.) ESWC 2017. LNCS, vol. 10249, pp. 554–570. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58068-5_34
18. Musto, C., Lops, P., Basile, P., de Gemmis, M., Semeraro, G.: Semantics-aware graph-based recommender systems exploiting linked open data. In: Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization, pp. 229–237. ACM (2016)
19. Musto, C., Narducci, F., Lops, P., De Gemmis, M., Semeraro, G.: ExpLOD: a framework for explaining recommendations based on the linked open data cloud. In: Proceedings of the 10th ACM Conference on Recommender Systems, pp. 151–154. RecSys 2016. ACM, New York (2016)
20. Musto, C., Semeraro, G., de Gemmis, M., Lops, P.: Tuning personalized pagerank for semantics-aware recommendations based on linked open data. In: Blomqvist, E., Maynard, D., Gangemi, A., Hoekstra, R., Hitzler, P., Hartig, O. (eds.) ESWC 2017. LNCS, vol. 10249, pp. 169–183. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58068-5_11
21. Nguyen, P., Tomeo, P., Di Noia, T., Di Sciascio, E.: An evaluation of SimRank and personalized PageRank to build a recommender system for the web of data. In: Proceedings of the 24th WWW, pp. 1477–1482. ACM (2015)
22. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. Proc. IEEE **104**(1), 11–33 (2016)
23. Nickel, M., Tresp, V., Kriegel, H.P.: Factorizing yago: scalable machine learning for linked data. In: Proceedings of the 21st International Conference on World Wide Web, pp. 271–280. ACM (2012)
24. Noia, T.D., Ostuni, V.C., Tomeo, P., Sciascio, E.D.: Sprank: semantic path-based ranking for top-n recommendations using linked open data. ACM Trans. Intell. Syst. Technol. (TIST) **8**(1), 9 (2016)
25. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowl. Data Eng. **22**(10), 1345–1359 (2010)
26. Passant, A.: dbrec — music recommendations using DBpedia. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010. LNCS, vol. 6497, pp. 209–224. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17749-1_14
27. Piao, G., Breslin, J.G.: Measuring semantic distance for linked open data-enabled recommender systems. In: Proceedings of the 31st Annual ACM Symposium on Applied Computing, pp. 315–320. ACM, Pisa (2016)
28. Piao, G., Breslin, J.G.: Factorization machines leveraging lightweight linked open data-enabled features for top-N recommendations. In: Bouguettaya, A., Gao, Y., Klimenko, A., Chen, L., Zhang, X., Dzerzhinskiy, F., Jia, W., Klimenko, S.V., Li, Q. (eds.) WISE 2017. LNCS, vol. 10570, pp. 420–434. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68786-5_33

29. Rendle, S.: Factorization machines with libFM. ACM Trans. Intell. Syst. Technol. **3**(3), 57:1–57:22 (2012)
30. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, pp. 452–461. AUAI Press (2009)
31. Rendle, S., Schmidt-Thieme, L.: Pairwise interaction tensor factorization for personalized tag recommendation. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 81–90. ACM (2010)
32. Sheth, A., Perera, S., Wijeratne, S., Thirunarayan, K.: Knowledge will propel machine understanding of content: extrapolating from current examples. In: 2017 IEEE/WIC/ACM International Conference on Web Intelligence. ACM (2017)
33. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014)
34. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI, pp. 1112–1119 (2014)
35. Zhang, F., Yuan, N.J., Lian, D., Xie, X., Ma, W.Y.: Collaborative knowledge base embedding for recommender systems. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2016, pp. 353–362. ACM, New York (2016)