Bridging the Gap between Linked Open Data-based Recommender Systems and Distributed Representations

Pierpaolo Basile[a,*], Claudio Greco[a], Alessandro Suglia[a], Giovanni Semeraro[a]

[a] *Department of Computer Science, University of Bari Aldo Moro, Via E. Orabona, 4 - 70125 Bari, Italy*

Abstract

Recently, several methods have been proposed for introducing Linked Open Data (LOD) into recommender systems. LOD can be used to enrich the representation of items by leveraging RDF statements and adopting graph-based methods to implement effective recommender systems. However, most of those methods do not exploit embeddings of entities and relations built on knowledge graphs, such as datasets coming from the LOD. In this paper, we propose a novel recommender system based on holographic embeddings of knowledge graphs built from Wikidata, a free and open knowledge base that can be read and edited by both humans and machines. The evaluation performed on three standard datasets such as Movielens 1M, Last.fm and LibraryThing shows promising results, which confirm the effectiveness of the proposed method.

Keywords: Recommender Systems, Knowledge Graph Embedding, Linked Data

1. Introduction

The Linked Open Data (LOD) [1] was launched in 2007 to support the publishing of data in RDF[1] format adopting shared vocabularies. The Linked

*Corresponding author

Email addresses: pierpaolo.basile@uniba.it (Pierpaolo Basile), claudiogaetanogreco@gmail.com (Claudio Greco), alessandro.suglia@gmail.com (Alessandro Suglia), giovanni.semeraro@uniba.it (Giovanni Semeraro)

[1] http://www.w3.org/TR/rdf-concepts/

DOI: 10.1016/j.is.2019.07.001

Open Data (LOD) cloud forms a knowledge base which covers several domains, ranging from geographical data to information about media (movies, books, etc.). Currently, the LOD cloud contains almost 150 billions of RDF triples and consists of about 10,000 datasets[2]. This huge amount of machine-readable knowledge can be exploited to improve knowledge-intensive applications, such as recommender systems. Recently, several methods have been proposed to introduce LOD information into recommender systems [2, 3, 4, 5, 6, 7, 8]. Basically, item descriptions can be enriched by leveraging relations described into the LOD cloud. For example, the movie *E. T. the Extra-Terrestrial*, identified by a specific Wikidata URI[3], can be described by the following properties: *director:Steven Spielberg*, *composer:John Williams* and *distributor:Universal Studios*. In addition, the availability of Linked Data allows to discover interesting relations between movies by following their properties. For example, by following the relation *composer:John Williams*, we can discover that *John Williams* was the music composer of *Star Wars Rebels* and *Lost in Space*.

Recently, vector space embeddings of knowledge graphs have gained considerable attention [9] and have been applied to several tasks such as link prediction, entity disambiguation, extraction of taxonomies and question answering [10, 11, 12, 13]. These approaches allow to represent entities and relations through an embedding, which is a continuous vector representation able to capture the semantics of an entity or relation. In this work we investigate holographic embeddings (HolE) [14], which exploit the circular correlation of entity embeddings to create compositional representations of binary relational data coming from the LOD cloud. By exploiting HolE, we model a recommendation framework in which items are represented by entity embeddings and the profile of the target user is built by composing entity embeddings related to items already rated by the user. The recommendation is performed by evaluating the similarity between the user profile and the embeddings associated to the

[2] `http://stats.lod2.eu/`

[3] `http://www.wikidata.org/entity/Q11621`

items not rated by the user. We evaluate the proposed framework in a top-n recommendation scenario by comparing it with several baselines.

The main research question of our work is to prove the ability of knowledge graph embeddings to represent items in a Content-based Recommender System (CBRS). Their relevance is justified by the fact that the user profile can be built very efficiently by exploiting pre-trained embeddings and the recommendation phase involves vector operations which can be greatly speeded up by using dedicated architectures (e.g., GPUs).

The paper is organized as follows: Section 2 reports the current literature about LOD-based recommender systems, while Section 3 describes the adopted methodology. Evaluation and results are provided in Section 4, while conclusions and future work close the paper.

2. Related Work

In the last years, several approaches for introducing LOD into recommender systems have been proposed. LOD-based recommender systems have their roots in ontology-based recommender systems [15], while a first attempt to exploit LOD to compute semantic similarity between items by using DBpedia [16] was proposed by Passant [17]. DBpedia is the RDF mapping of Wikipedia and it is the core of the LOD cloud. Other papers investigated the role of DBpedia in computing semantic similarity between items. In [2] the computation of semantic similarity in DBpedia is exploited to produce personalized music playlists, while in [3] a similarity measure inspired by Information Theory and adapted to the scenario of LOD is used to compute the similarity between items in a collaborative recommendation approach. Datasets in the LOD cloud can be used as data sources to enrich the representation of both items and the user profile. For example, in [4] DBpedia is used to retrieve one or more genres played by each artist extracted from Facebook. The retrieved genres are used to find more

artists in DBpedia for providing suggestions. Similarly in [18] Freebase[4], a large collaborative knowledge base, is exploited for describing artists, while in [19] LinkedGeoData[5] is used to extract features for describing point of interests.

In the previously mentioned papers, LOD are used to cope the problem of *limited content analysis* [20], which affects content-based recommendation approaches. On the other hand, several papers have investigated the impact of the use of LOD features on recommender systems performance. A relevant paper in this direction is [5], which analyses the use of manually selected properties in the context of movie recommendation. The work presented in [21] investigates the impact of LOD features on two types of recommendation techniques: PageRank and text classification models. The reported results prove the effectiveness of introducing LOD features, as confirmed by further work in several domains, such as event recommendation [22], book recommendation [23] and e-learning resources recommendation [24]. In 2014, during the ESWC 2014 Recommender Systems Challenge[6], several recommendation approaches based on LOD were proposed. The best system [25, 26] aggregates several approaches, such as Random Forests, Logistic Regression and PageRank with Priors, leveraging a diverse sets of features retrieved from the LOD cloud. An interesting approach to automatically select relevant features from the LOD was described in [6, 7], in which the authors applied several feature selection strategies to find the best set of LOD features for describing items. In [8] a hybrid approach, which combines collaborative features with graph-based ones extracted from the LOD, is used to perform sound and music recommendations.

All the above mentioned papers do not exploit entity and relation embeddings built from a knowledge graph. Recently, embeddings of knowledge graphs have been exploited in several tasks, obtaining promising results [9, 27, 28]. In this paper, we propose a recommendation framework based on graph em-

[4] `https://www.freebase.com/`

[5] `http://linkedgeodata.org`

[6] http://2014.eswc-conferences.org/important-dates/call-RecSys

beddings built from Wikidata[7]. To the best of our knowledge, this is the first attempt to exploit Wikidata to build entity and relation embeddings in recommender systems.

3. Methodology

In this section we describe our approach for computing a user profile based on knowledge base embeddings. First, in sub-section 3.1 we introduce the general compositional vector space framework for knowledge graphs, next in sub-section 3.2 we explain a specific instance of this class of methods called Holographic Embeddings (HolE), last in sub-section 3.3 we define our approach to solve the top-n recommendation task which exploits HolE in a CBRS to build a user profile from user preferences.

3.1. Basics of Knowledge Graphs Embeddings

A generic knowledge graph can be described by a set of entities E and a set of predicates P. Given a predicate $p \in P$, we can define a binary relation $\mathcal{R}_p \subseteq E \times E$, which is intended as the set of all pairs of entities related by the predicate p. For each pair of entities, the characteristic function $\phi_p : E \times E \rightarrow \{\pm 1\}$ indicates whether it is an element of $\mathcal{R}_p$. An element $\mathcal{R}_p(s, o)$ is called a triple and is composed by a subject s and an object o related by p, where $s, o \in E$.

By exploiting compositional vector space models, it is possible to learn the characteristic functions of the relations between entities in a knowledge graph, casting the learning task as a supervised learning problem. In particular, these models should be able to estimate the conditional probability $\Pr(\phi_p(s, o) = 1|\Theta)$ directly from the relations in the knowledge graph, where Θ denotes the set of all embeddings. A specific formulation of this kind of models can be described

[7] https://www.wikidata.org/

as follows:

$$\Pr(\phi_p(s,o)=1|\Theta)=\sigma(\eta_{spo})=\sigma(\boldsymbol{r}_p^{\intercal}(\boldsymbol{e}_s\circ e_o)), \tag{1}$$

where $\boldsymbol{r}_p \in \mathbb{R}^{d_r}$ is obtained by a lookup operation on a relation embedding matrix $\boldsymbol{W}_r \in \mathbb{R}^{|R|\times d_r}$, $\boldsymbol{e}_s, \boldsymbol{e}_o \in \mathbb{R}^{d_e}$ are obtained by a lookup operation on an item embedding matrix $\boldsymbol{W}_e \in \mathbb{R}^{|R|\times d_e}$, $\Theta=\{\boldsymbol{W}_r, \boldsymbol{W}_e\}$, $\sigma(x)=1/(1+\exp(-x))$ denotes the logistic function, $\circ : \mathbb{R}^{d_e}\times\mathbb{R}^{d_r}\rightarrow\mathbb{R}^{d_p}$ is the compositional operator in the defined vector space able to create a vector representation for the pair (s,o) from the embeddings $\boldsymbol{e}_s, \boldsymbol{e}_o$; d_e is the size of the entity embeddings, d_r is the size of the relation embeddings and d_p is the size of the embeddings obtained by the compositional operator.

Given a dataset $D=\{(x_i,y_i)\}_{i=1}^m$ of m existing and non-existing relation instances, where $x_i \in P\times E\times E$ denotes a triple and $y_i \in \{\pm 1\}$ denotes its label, we want to learn vector representations of entities and relations Θ such that the Equation 1 best approximates the value of the characteristic function for all the examples in the dataset D. For instance, this can be done by minimizing the following pairwise ranking loss:

$$L(D,\Theta)=\sum_{i\in D^+}\sum_{j\in D^-}\max(0,\gamma+\sigma(\eta_j)-\sigma(\eta_i)), \tag{2}$$

where D^+ denotes the set of existing triples, D^- denotes the set of non-existing triples and $\gamma>0$ specifies the width of the margin, as in [27]. In this way, the model learns to rank the existing triples higher than the non-existing ones.

3.2. HolE

In this work we exploit Holographic Embeddings (HolE) [14], which is a specific instance of the compositional vector space framework presented in the previous section to learn holographic embeddings for entities and relations in a knowledge graph. HolE uses the *circular correlation* operator $\star$ (in place of the

$\circ$ operator), defined as follows:

$$[\boldsymbol{a} \star \boldsymbol{b}]_k = \sum_{i=0}^{d-1} a_i b_{(k+i) \bmod d}. \tag{3}$$

Compositional vector space models equipped with the circular correlation operator have obtained superior performance over other methods presented in the literature, as demonstrated in [14]. The use of the circular correlation has several advantages, such as:

- it allows to learn vector representations for the relations which encode semantically similar interactions between the entities which take part in the relations;
- it is non commutative, thus it is able to model asymmetric relations in knowledge graphs;
- the component $[\boldsymbol{a} \star \boldsymbol{b}]_0 = \sum_i a_i b_i$ corresponds to the dot product $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$, which allows to take into account the similarity between entities.

The model capability to estimate the probability defined in Equation 1 represents an appealing property in scenarios in which we want to understand if it is possible to relate two entities of the knowledge graph. Particularly, this is really important for the knowledge completion task which could be a way to extend the relations in a knowledge base. By solving this task, the entity representation encodes similarities between entities which take part in similar relations.

3.3. Exploiting HolE in a Content-based Recommender System

Given a set R of user preferences (u, i, r), where $u \in U$ is the user identifier, $i \in I$ is the item identifier and $r \in \{0, 1\}$ is the binary preference of the user u related to the item i, the aim of a CBRS is to learn a user profile for each user u by leveraging the item representations and then to exploit it for providing a list of suggestions ranked according to user preferences.

In order to generate the item representation, we exploit a knowledge graph which represents the information related to the domain in which the recommender system will be evaluated (i.e., music, movies, etc.). Obviously, we assume that the considered knowledge graph contains information associated to the set of items I. The knowledge graph triples can be used as a training set for the HolE method to learn representations for each entity by minimizing the loss function reported in Equation 2 through the Stochastic Gradient Descent (SGD), where $\circ = \star$.

We define two strategies that can be exploited to obtain refined representations associated to each entity of the knowledge graph:

L2 normalization: we apply L_2 normalization on the $\boldsymbol{W}_e$ matrix obtaining the matrix $\tilde{\boldsymbol{W}}_e$;

PCA-based: inspired by the word embeddings preprocessing strategy extensively evaluated in the work presented in [29], we apply it to the entity embedding matrix $\boldsymbol{W}_e$. First, we center the matrix $\boldsymbol{W}_e$ to its mean and then we compute Principal Component Analysis (PCA) selecting the first $d_e/100$ components that we use to project the original representations according to the selected components obtaining the matrix $\tilde{\boldsymbol{W}}_{\boldsymbol{e}}$.

The *L2 normalization* strategy is applied to the reference vector space in which the embeddings lies on, while the *PCA-based* strategy removes the nonzero mean vector from all embeddings and projects the representations away from the dominating directions. While the latter seems a counter-intuitive preprocessing procedure, it has the advantage of yielding *"purified"* entity representations, as demonstrated in the experiments reported in [29].

The recommendation process can be computed by using two different strategies: the first uses distributed representations only, while the second uses item embeddings to feed a classifier.

Distributed representations: The matrix $\tilde{\boldsymbol{W}}_e$ contains the embeddings associated to each entity of the knowledge graph. It is possible to extract the embedding associated to each item $i \in I$ from it by selecting the related

row. We denote the set of item embeddings as I_e, for each item $i \in I$. Given $i \in I$, we denote as $v(i)$ the embedding of i. HolE exploits the computed item representations to generate the user profile of a given user u considering his/her positive preferences $I^+(u) = \{i \in I \mid \exists(u, i, r) \in R \wedge r == 1\}$ and negative preferences $I^-(u) = \{i \in I \mid \exists(u, i, r) \in R \wedge r == 0\}$. We compute the centroid of the set of embeddings contained in $\tilde{\boldsymbol{W}}_e$ as follows:

$$\boldsymbol{c}_e = \frac{1}{|E|} \sum_{k \in E} \tilde{\boldsymbol{W}}_{\boldsymbol{e}}(k), \tag{4}$$

where $\tilde{\boldsymbol{W}}_{\boldsymbol{e}}(k)$ denotes the k-th row of the entity embedding matrix $\tilde{\boldsymbol{W}}_{\boldsymbol{e}}$. We evaluate the positive user profile $\boldsymbol{u}^+$ as follows:

$$\boldsymbol{u}^+ = \sum_{i \in I^+(u)} v(i) - \boldsymbol{c}_e \tag{5}$$

and we exploit it to generate the user profile $\boldsymbol{u}$ by an orthogonalization procedure taking into account the negative user preferences $I^-(u)$. In a geometric space the concept of relevance is expressed in terms of similarity, while the concept of irrelevance is defined by orthogonality (similarity equals to zero). Given two vectors a and b in a vector space V endowed with a scalar product, $a \quad NOT \quad b$ corresponds to the projection of a onto the orthogonal space $\langle b \rangle^{\perp} \equiv \{v \in V : \forall b \in \langle b \rangle, v \cdot b = 0\}$, where $\langle b \rangle$ is the subspace $\{\lambda b : \lambda \in \mathbb{R}\}$. The negation operator is implemented using the Gram-Schmidt orthogonalization procedure on the set of vectors $\langle \boldsymbol{i}_1, \boldsymbol{i}_2, \ldots, \boldsymbol{i}_{|I^-(u)|}, \boldsymbol{u}^+ \rangle$ obtaining the user profile $\boldsymbol{u}$ as the last vector of the resulting set of vectors. In this way, we are able to remove from the positive user profile vector $\boldsymbol{u}^+$ all the components related to the negative item embeddings. Two particular cases may happen when building the user profile which are described as follows:

- if $I^+(u)$ is empty, we consider as the positive user profile $\boldsymbol{u}^+$ the centroid of the vector space $\boldsymbol{c}_e$ and we apply the orthogonalization procedure as described before;
- if $I^-(u)$ is empty, we consider as the user profile $\boldsymbol{u}$ only the positive user profile vector $\boldsymbol{u}^+$.

The recommendation process for a given user u is completed by evaluating a list of n items ranked in descending order according to the cosine similarity between the user profile $\boldsymbol{u}$ and the item embedding $\boldsymbol{i}$ of each item $i \in I \setminus I(u)$.

Classifiers: Given the capability of distributed representations to capture latent factors among dataset instances, we decide to feed a classifier with the learned item embeddings in order to learn user profiles. In this scenario, each dimension of the embeddings is a feature exploited by the classifier. In particular, we create a dataset composed by item embeddings associated to the rated items contained in $I^+(u)$ and $I^-(u)$ for the given user u. We exploit a classifier to estimate the probability $P(i|u)$ of a like given to the item i by the user u to generate a ranked list of suggestions sorted in descending order. Two particular cases may happen when building the user profile which are described as follows:

- if $I^+(u)$ is empty, we add to the dataset a positive instance generated by orthogonalizing the centroid $\boldsymbol{c}_e$ with respect to the centroid of the negative items, built using the same procedure of $\boldsymbol{u}^+$. The idea is that all the items not explicitly rated as negative can be positive;
- if $I^-(u)$ is empty, we add to the dataset a positive instance generated by orthogonalizing the centroid $\boldsymbol{c}_e$ with respect to the centroid of the positive items $\boldsymbol{u}^+$. The idea is that all the items not explicitly rated as positive can be negative.

4. Experimental Evaluation

In this section, we present the experimental evaluation designed to assess the effectiveness of the proposed method by comparing its performance on the top-n recommendation task with state-of-the-art baselines, such as classical collaborative filtering techniques and matrix factorization algorithms. All the algorithms involved in the experimental evaluation should be able to leverage binary user feedback to generate appropriate lists of suggestions. In sub-section 4.1 we describe the experimental design and the datasets used in the evaluation, while in sub-section 4.2 we discuss the experimental results.

4.1. Experimental Design

Datasets: Experiments were performed against three state-of-the-art datasets: Movielens 1M (ML1M)[8], Last.fm[9] and LibraryThing[10]. ML1M is a well-known dataset related to the movies domain, Last.fm is a dataset which describes user listening preferences towards specific artists or music bands whereas LibraryThing is a rating dataset of books extracted from the website LibraryThing.

Protocol: The fundamental requirement for the experimental evaluation is a mapping between the items in the dataset and specific identifiers in a knowledge base. In this work, we decided to exploit Wikidata[11] as knowledge base from which triples related to the specific application domain of the considered datasets were extracted. For the ML1M dataset, we retrieved all the triples involving properties reported in Table 2 for the movies which are instance of *wd:Q11424*[12] *(film)* or of one of its subclasses using the property *wdt:P31*[13] *(instance of)*. Precisely, we considered movies which are instance of a type whose depth in the class hierarchy is at most four starting from *wd:Q11424 (film)* as shown in the SPARQL query 1.

For the Last.fm dataset we retrieved all the triples involving the properties reported in Table 3 for the entities corresponding to artists or bands identified according to their occupation (represented by the property *wdt:P106*). In particular, artists are entities whose occupation is *wd:Q177220 (singer)*, *wd:Q639669 (musician)* or *wd:Q36834 (composer)*, while bands are entities whose occupation is *wd:Q215380 (band)*. Precisely, we considered entities whose occupation is instance of a type whose depth in the class hierarchy is at most four starting from the relative most general type. The queries employed to retrieve artist and band properties are shown in Listings 2 and 3, respectively.

For the LibraryThing dataset, we exploited a similar procedure to the one

[8] `http://grouplens.org/datasets/movielens/`
[9] `https://grouplens.org/datasets/hetrec-2011/`
[10] `http://www.macle.nl/tud/LT`
[11] `https://www.wikidata.org/`
[12] The prefix *wd* stands for `http://www.wikidata.org/entity/`
[13] The prefix *wdt* stands for `http://www.wikidata.org/prop/direct/`

adopted for ML1M. In particular, we used as a reference type for books the entity represented by the identifier *wd:Q571 (book)* and for each instance of this type (or one of its subclasses) we retrieved all the values associated to the properties reported in Table 4. The retrieved triples are stored in a *Sleepycat Berkeley DB* database provided by the *RDFlib* library[14]. Table 1 shows some statistics related to the retrieved triples.

We used the ML1M, Last.fm and LibraryThing DBpedia mappings, provided by [30], to find the corresponding Wikidata URIs by querying the DBpedia knowledge base using the SPARQL query 4.

	ML1M	Last.fm	LibraryThing
Triples	1,930,649	508,478	336,511
Entities	395,813	225,889	152,162
Predicates	24	14	13

Table 1: Statistics about the retrieved triples.

We filtered out items from the original datasets that have not a corresponding identifier in the Wikidata knowledge base or have no triples describing them in the *Sleepycat Berkeley DB* database. Statistics of the filtered datasets are reported in Table 5.

The experimental evaluation requires that the datasets contain binary user preferences, so we applied a binarization procedure to the datasets. For the ML1M dataset we considered as positive ratings those that were greater than 3, as negatives all the others. For the Last.fm dataset we considered as positive ratings those that were greater than the median of the users listening count, negative otherwise. Finally, for the LibraryThing dataset the binarization procedure was not required because it already contains binary feedback.

We applied a 5-fold cross validation exploiting the folds obtained by using

[14] `https://github.com/RDFLib/rdflib`

the script[15] of the *RiVal* evaluation toolkit [31]. In order to make our experiments reproducible, all the metrics are calculated by using *RiVal* following the *TestRatings* strategy [32]. The final F1@K measure for each algorithm is computed by averaging the F1@K measure obtained on each fold.

Baselines: In order to confirm the effectiveness of our approach, we compared it with several state-of-the-art techniques, as *collaborative filtering* without side information and with side information (`BPRMF-LOD`).

Popularity a non-personalized technique which recommends most-popular items to users;

U2U k-nearest neighbour user-based collaborative filtering [33], where the preference estimation is computed according to the preference expressed by users similar to the one to whom the suggestions will be generated;

I2I k-nearest neighbour item-based collaborative filtering [34], uses similarities between the rating patterns of items to estimate the preference of a given user for a given item;

BPRMF a matrix factorization model for item recommendation based on *Bayesian Personalized Ranking optimization criterion (BPR-Opt)* [35];

BPRMF-LOD a configuration of the BPRMF model able to exploit the LOD features retrieved from the knowledge base associated to the items as side information. Side information are represented as attributes associated to each item. Attributes are obtained by concatenating the property and the subject of each triple associated to the item, for example the item *E.T._the_Extra-Terrestrial* has the attribute *director_Steven_Spielberg*;

WRMF a weighted matrix factorization algorithm based on the *Alternating Least Squares (ALS)* learning method [36];

[15] `https://github.com/recommenders/rival/`

BPRSlim *Sparse Linear Methods (SLIM)* for item recommendation using *BPR-Opt* [37].

The adopted baselines are available in the *MyMediaLite* recommender system library[16]. For the `BPRMF-LOD` configuration, we specified LOD features using the *–item-attributes* parameter.

Overview of the parameters: The `HolE` model was trained by using *Adagrad* optimizer [38] for at most 500 epochs with a learning rate of 0.1. The number of batches was set to 100, the embedding size was fixed to 300 and the margin for the pairwise ranking loss was set to 0.2. For what concerns the baseline parameters, `I2I` and `U2U` are evaluated by setting the neighbourhood size to 30, 50 and 80, while the matrix factorization algorithms are run by learning 10, 30 and 50 latent factors.

Model implementation details: The model was implemented in *Python 3* by leveraging the *HolE* implementation provided by the *scikit-kge* library[17] and *NumPy*[18] and *scikit-learn*[19] libraries. The method (**HolE-LR**) that exploits items' embeddings as features in a classifier is based on the Logistic Regression implementation provided by the *scikit-learn* library.

4.2. Discussion of the results

The results of the experimental evaluation for the datasets ML1M, Last.fm and LibraryThing are reported in Tables 6, 7 and 8, respectively. The best-performing baseline is reported in italics, while the overall best configuration is highlighted in bold.

Different configurations of the *HolE* approach have been evaluated. In particular, we denote as `HolE-RT` the approach which applies orthogonalization and removal of the centroid, as `HolE-R` the approach which applies the removal of the centroid only, as `HolEE-T` the approach which applies orthogonalization only and

[16] `http://www.mymedialite.net/documentation/item_prediction.html`
[17] `https://github.com/mnick/scikit-kge`
[18] `http://www.numpy.org/`
[19] `http://scikit-learn.org/`

as HolE the approach which applies neither of the two. Four particular configurations of HolE (`HolE-RT-5`, `HolE-RT-10`, `HolE-PCA-RT-5`, `HolE-PCA-RT-10`) exploit a limited number of negative ratings, respectively 5 and 10. We apply this strategy because, when a user has a large number of negative ratings, the orthogonalization process causes underflow errors. The `HolE-LR` is the approach based on Logistic Regression.

As shown by the experimental evaluation, the `HolE` configurations obtain results comparable to the best baseline according to the adopted evaluation metrics. Moreover, HolE is able to outperform all the baselines for both F1@10 and F1@15 in the Last.fm dataset. `HolE-LR` is able to overcome all the baselines in the LibraryThing dataset. The results of the experimental evaluation are validated using the Wilcoxon signed-rank test with a significance level $\alpha = 0.05$. For the ML1M dataset, the statistical significance tests highlight that the differences between `HolE-LR` and `WRMF-10` are statistically significant on the top-5 and top-15, but not on the top-10, while the differences between `BPRMF-LOD-10` and `HolE-PCA-RT-10` are statistically significant with respect to all the cutoffs. For the Last.fm dataset, the differences between `HolE` and `BPRMF-LOD-10` are statistically significant with respect to all the F1 cutoffs. For the LibraryThing dataset, the differences between `HolE-LR` and `BPRMF-LOD-10` are statistically significant with respect to all the F1 cutoffs.

Regarding the experimental evaluation on the Last.fm dataset, we can notice that the results for the baselines and the `HolE` configurations are very similar to each other. We think that the similarity of the results is probably caused by the low number of ratings per user contained in the test set. Indeed, the same behaviour is not observed in the experimental evaluation on both ML1M and LibraryThing datasets.

In addition to the good performance, it is worth noting that all the `HolE` configurations (except `HolE-LR`) are able to effectively deal with the `new user problem` [39] because they build the user profile without requiring a costly offline training procedure like in matrix factorization techniques. However, if a new item is added to the catalogue a new training procedure of the knowledge graph

embeddings and the computation of the PCA on the embeddings matrix (for `HolE-PCA` only) is required. Finally, it is important to underline that `HolE` is completely content-based and even when users have few ratings, as in the Last.fm dataset, it is able to achieve the best performance.

The fact that HolE is able to achieve performance close to collaborative filtering approaches is encouraging since it exploits only information about items content without any knowledge about other users. This allows to build transparent approaches able to provide an explanation about the provided recommendation by exploiting the description associated to the suggested item [40]. We plan to investigate this aspect in the future.

5. Conclusion and Future Work

In this paper, we propose a Content-based Recommender System that exploits knowledge graph embeddings for representing items. The embeddings are built by leveraging on triples extracted from Wikidata. Several approaches for building the user profiles and for generating a list of suggestions for the user are proposed. The evaluation performed on three datasets such as Movielens 1M, Last.fm and LibraryThing proves the effectiveness of our approach in achieving performance which is comparable to the performance of state-of-the-art collaborative systems and which, in some cases, outperforms the performance of all the baselines. This is an encouraging outcome since our approach exploits only the item description without any knowledge about other users.

This outcome opens several perspectives for further investigations:

- it is possible to include in the knowledge graph information about users, items and ratings. By adding triples of the type *<userId> rates <itemId>*, it is possible to learn embeddings which can be exploited to: 1) build an embedding for each user; 2) compute the similarity between users and items; 3) try to predict the probability of a link between an user and any item. It is important to underline that 3) allows to implement a recommender system able to rank items according to their probability;

- using the probability of a link between an item and any other node in the graph, we can predict the most likely propriety that links them. This approach could be useful to explain the recommendation;

- HolE is able to build embeddings related to properties. This allows to predict the most related properties associated to any node embedding. If we consider the user vector profile as an embedding, we can predict the most appropriate properties that describe the user profile. These properties can be exploited to provide a transparent description of the user profile in according to the European Union's new General Data Protection Regulation;

- embeddings can be used to initialize the weights of a deep neural network in order to implement recommender systems based on deep learning techniques. The promising results obtained by `HolE-LR` justify the use of classifiers to obtain higher performance.

Currently, our approach based on knowledge graph embeddings is not able to clearly outperform collaborative baselines. However, its potentialities are remarkable and need further investigations which may lead to more transparent and effective recommender systems.

Appendix A - SPARQL queries

```
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
PREFIX wd: <http://www.wikidata.org/entity/>

SELECT ?s ?p ?o WHERE {
{
?s wdt:P31 ?type.
?type wdt:P279 ?t1.
?t1 wdt:P279 ?t2.
?t2 wdt:P279 ?t3.
?t3 wdt:P279 movie_type.
}
```

```
UNION
{
?s wdt:P31 ?type.
?type wdt:P279 ?t1.
?t1 wdt:P279 ?t2.
?t2 wdt:P279 movie_type.
}
UNION
{
?s wdt:P31 ?type.
?type wdt:P279 ?t.
?t wdt:P279 movie_type.
}
UNION
{
?s wdt:P31 ?type.
?type wdt:P279 movie_type.
}
UNION
{
?s wdt:P31 movie_type.
}
VALUES ?p {movie_properties}
?s ?p ?o.
}
```

Listing 1: SPARQL query used to retrieve all the properties movie_properties associated to the entities instance of movie_type.

```
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
PREFIX wd: <http://www.wikidata.org/entity/>

SELECT ?s ?p ?o WHERE {
{
?s wdt:P106 ?type.
?type wdt:P279 ?t1.
?t1 wdt:P279 ?t2.
?t2 wdt:P279 ?t3.
?t3 wdt:P279 artist_type.
```

```
}
UNION
{
?s wdt:P106 ?type.
?type wdt:P279 ?t1.
?t1 wdt:P279 ?t2.
?t2 wdt:P279 artist_type.
}
UNION
{
?s wdt:P106 ?type.
?type wdt:P279 ?t.
?t wdt:P279 artist_type.
}
UNION
{
?s wdt:P106 ?type.
?type wdt:P279 artist_type.
}
UNION
{
?s wdt:P106 artist_type.
}
VALUES ?p {artist_properties}
?s ?p ?o.
}
```

Listing 2: SPARQL query used to retrieve all the properties artist_properties associated to the artists instance of artist_type.

```
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
PREFIX wd: <http://www.wikidata.org/entity/>

SELECT ?s ?p ?o WHERE {
{
?s wdt:P31 ?type.
?type wdt:P279 ?t1.
?t1 wdt:P279 ?t2.
?t2 wdt:P279 ?t3.
```

```
?t3 wdt:P279 band_type.
}
UNION
{
?s wdt:P31 ?type.
?type wdt:P279 ?t1.
?t1 wdt:P279 ?t2.
?t2 wdt:P279 band_type.
}
UNION
{
?s wdt:P31 ?type.
?type wdt:P279 ?t.
?t wdt:P279 band_type.
}
UNION
{
?s wdt:P31 ?type.
?type wdt:P279 band_type.
}
UNION
{
?s wdt:P31 band_type.
}
VALUES ?p {band_properties}
?s ?p ?o.
}
```

Listing 3: SPARQL query used to retrieve all the properties band_properties associated to the entities instance of band_type.

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
SELECT ?wikidata_uri WHERE {
dbpedia_uri owl:sameAs ?wikidata_uri
FILTER(REGEX(?wikidata_uri, "www.wikidata.org" ))
}
```

Listing 4: SPARQL query used to find the Wikidata URI corresponding to the DBpedia URI, where dbpedia_uri is the item identifier in DBpedia knowledge base.

References

[1] C. Bizer, The emerging web of linked data, IEEE Intelligent Systems 24 (5) (2009) 87–92.

[2] C. Musto, G. Semeraro, P. Lops, M. De Gemmis, F. Narducci, Leveraging social media sources to generate personalized music playlists, in: International Conference on Electronic Commerce and Web Technologies, Springer, 2012, pp. 112–123.

[3] R. Meymandpour, J. G. Davis, Enhancing recommender systems using linked open data-based semantic analysis of items, in: Proceedings of the 3rd Australasian Web Conference (AWC 2015), Vol. 27, 2015, p. 30.

[4] S. Bostandjiev, J. O'Donovan, T. Höllerer, Tasteweights: A visual interactive hybrid recommender system, in: Proceedings of the Sixth ACM Conference on Recommender Systems, RecSys '12, ACM, New York, NY, USA, 2012, pp. 35–42.

[5] T. Di Noia, R. Mirizzi, V. C. Ostuni, D. Romito, Exploiting the web of data in model-based recommender systems, in: Proceedings of the Sixth ACM Conference on Recommender Systems, RecSys '12, ACM, New York, NY, USA, 2012, pp. 253–256.

[6] C. Musto, P. Lops, P. Basile, M. de Gemmis, G. Semeraro, Semantics-aware graph-based recommender systems exploiting linked open data, in: Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization, ACM, 2016, pp. 229–237.

[7] C. Musto, P. Basile, P. Lops, M. de Gemmis, G. Semeraro, Introducing linked open data in graph-based recommender systems, Information Processing and Management 53 (2) (2017) 405 – 435.

[8] S. Oramas, V. C. Ostuni, T. D. Noia, X. Serra, E. D. Sciascio, Sound and music recommendation with knowledge graphs, ACM Transactions on Intelligent Systems and Technology (TIST) 8 (2) (2016) 21.

[9] M. Nickel, K. Murphy, V. Tresp, E. Gabrilovich, A review of relational machine learning for knowledge graphs, Proceedings of the IEEE 104 (1) (2016) 11–33.

[10] Y. Lin, Z. Liu, M. Sun, Y. Liu, X. Zhu, Learning entity and relation embeddings for knowledge graph completion., in: AAAI, Vol. 15, 2015, pp. 2181–2187.

[11] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph embedding by translating on hyperplanes., in: AAAI, Vol. 14, 2014, pp. 1112–1119.

[12] H. Xiao, M. Huang, X. Zhu, Transg: A generative model for knowledge graph embedding, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vol. 1, 2016, pp. 2316–2325.

[13] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph and text jointly embedding, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1591–1601.

[14] M. Nickel, L. Rosasco, T. A. Poggio, et al., Holographic embeddings of knowledge graphs., in: The Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16), 2016, pp. 1955–1961.

[15] S. E. Middleton, D. De Roure, N. R. Shadbolt, Ontology-based recommender systems, in: Handbook on ontologies, Springer, 2009, pp. 779–796.

[16] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, Dbpedia: A nucleus for a web of open data, in: The semantic web, Springer, 2007, pp. 722–735.

[17] A. Passant, dbrec–music recommendations using dbpedia, in: International Semantic Web Conference 2010 (ISWC-2010), Springer, 2010, pp. 209–224.

[18] S. Baumann, R. Schirru, Using linked open data for novel artist recommendations, in: 13th Internal Society for Music Information Retrival Conference, Porto, 2012.

[19] M. Schmachtenberg, T. Strufe, H. Paulheim, Enhancing a location-based recommendation system by enrichment with structured data from the web, in: Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14), ACM, 2014, p. 17.

[20] M. de Gemmis, P. Lops, C. Musto, F. Narducci, G. Semeraro, Semantics-aware content-based recommender systems, in: Recommender Systems Handbook, Springer, 2015, pp. 119–159.

[21] C. Musto, P. Basile, P. Lops, M. De Gemmis, G. Semeraro, Linked open data-enabled strategies for top-n recommendations., in: CBRecSys@ RecSys, 2014, pp. 49–56.

[22] H. Khrouf, R. Troncy, Hybrid event recommendation using linked data and user diversity, in: Proceedings of the 7th ACM Conference on Recommender Systems, RecSys '13, ACM, New York, NY, USA, 2013, pp. 185–192.

[23] L. Peska, P. Vojtas, Enhancing recommender system with linked open data, in: H. L. Larsen, M. J. Martin-Bautista, M. A. Vila, T. Andreasen, H. Christiansen (Eds.), Flexible Query Answering Systems: 10th International Conference, FQAS 2013, Granada, Spain, September 18-20, 2013. Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 483–494.

[24] S. Dietze, H. Drachsler, D. Giordano, A survey on linked data and the social web as facilitators for tel recommender systems, in: N. Manouselis, H. Drachsler, K. Verbert, O. C. Santos (Eds.), Recommender Systems for Technology Enhanced Learning: Research Trends and Applications, Springer New York, New York, NY, 2014, pp. 47–75.

[25] P. Basile, C. Musto, M. de Gemmis, P. Lops, F. Narducci, G. Semeraro, Aggregation strategies for linked open data-enabled recommender systems, in: European Semantic Web Conference, 2014.

[26] P. Basile, C. Musto, M. de Gemmis, P. Lops, F. Narducci, G. Semeraro, Content-based recommender systems + dbpedia knowledge = semantics-aware recommender systems, in: V. Presutti, M. Stankovic, E. Cambria, I. Cantador, A. Di Iorio, T. Di Noia, C. Lange, D. Reforgiato Recupero, A. Tordai (Eds.), Semantic Web Evaluation Challenge: SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers, Springer International Publishing, Cham, 2014, pp. 163–169.

[27] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: Advances in neural information processing systems, 2013, pp. 2787–2795.

[28] M. Nickel, X. Jiang, V. Tresp, Reducing the rank in relational factorization models by including observable patterns, in: Advances in Neural Information Processing Systems, 2014, pp. 1179–1187.

[29] J. Mu, S. Bhat, P. Viswanath, All-but-the-top: Simple and effective post-processing for word representations, arXiv preprint arXiv:1702.01417.
URL https://arxiv.org/abs/1702.01417

[30] T. D. Noia, V. C. Ostuni, P. Tomeo, E. D. Sciascio, Sprank: Semantic path-based ranking for top-n recommendations using linked open data, ACM Transactions on Intelligent Systems and Technology (TIST) 8 (1) (2016) 9.

[31] A. Said, A. Bellogín, Rival: a toolkit to foster reproducibility in recommender system evaluation, in: Proceedings of the 8th ACM Conference on Recommender systems, ACM, 2014, pp. 371–372.

[32] A. Bellogin, P. Castells, I. Cantador, Precision-oriented evaluation of recommender systems: an algorithmic comparison, in: Proceedings of the fifth ACM conference on Recommender systems, ACM, 2011, pp. 333–336.

[33] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl, Grouplens: an open architecture for collaborative filtering of netnews, in: Proceedings of

the 1994 ACM conference on Computer supported cooperative work, ACM, 1994, pp. 175–186.

[34] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th international conference on World Wide Web, ACM, 2001, pp. 285–295.

[35] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bpr: Bayesian personalized ranking from implicit feedback, in: Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, AUAI Press, 2009, pp. 452–461.

[36] Y. Hu, Y. Koren, C. Volinsky, Collaborative filtering for implicit feedback datasets, in: Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on, Ieee, 2008, pp. 263–272.

[37] X. Ning, G. Karypis, Slim: Sparse linear methods for top-n recommender systems, in: Data Mining (ICDM), 2011 IEEE 11th International Conference on, IEEE, 2011, pp. 497–506.

[38] J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, Journal of Machine Learning Research 12 (Jul) (2011) 2121–2159.

[39] J. B. Schafer, D. Frankowski, J. Herlocker, S. Sen, Collaborative filtering recommender systems, in: The adaptive web, Springer, 2007, pp. 291–324.

[40] C. Musto, F. Narducci, P. Lops, M. de Gemmis, G. Semeraro, Linked open data-based explanations for transparent recommender systems, International Journal of Human-Computer Studies.

Property	Description
wdt:P57	director
wdt:P58	screenwriter
wdt:P162	producer
wdt:P161	cast member
wdt:P344	director of photography
wdt:P262	production company
wdt:P136	genre
wdt:P921	main subject
wdt:P840	narrative location
wdt:P577	publication date
wdt:P495	country of origin
wdt:P364	original language of work
wdt:P166	award received
wdt:P1040	film editor
wdt:P86	composer
wdt:P1411	nominated for
wdt:P462	color
wdt:P2047	duration
wdt:P144	based on
wdt:P915	filming location
wdt:P2408	set period
wdt:P750	distributor
wdt:P941	inspired by
wdt:P179	series

Table 2: Wikidata properties associated to the items contained in the ML1M dataset.

	Property	Description
Artist	wdt:P21	sex or gender
	wdt:P27	country of citizenship
	wdt:P136	genre
	wdt:P272	production company
	wdt:P495	country of origin
	wdt:P166	award received
	wdt:P1411	nominated for
	wdt:P941	inspired by
Band	wdt:P527	has part
	wdt:P136	genre
	wdt:P495	country of origin
	wdt:P740	location of formation
	wdt:P737	influenced by
	wdt:P571	inception

Table 3: Wikidata properties associated to the artists and the bands contained in the Last.fm dataset.

Property	Description
wdt:P57	director
wdt:P136	genre
wdt:P50	author
wdt:P123	publisher
wdt:P495	country of origin
wdt:P364	original language of work
wdt:P840	narrative location
wdt:P674	characters
wdt:P155	follows
wdt:P156	followed by
wdt:P577	publication date
wdt:P571	inception
wdt:P110	illustrator
wdt:P166	award received

Table 4: Wikidata properties associated to the items contained in the LibraryThing dataset.

	ML1M	Last.fm	LibraryThing
Users	6,040	1,883	7,261
Items	3,227	8,674	9,418
Ratings	948,987	73,975	391,566
Sparsity	95.13%	99.55%	99.4%
Avg. ratings/user	157.12	39.28	53.927
Avg. positive ratings/user	89.92	19.63	34.964
Avg. negative ratings/user	67.19	19.66	18.963

Table 5: Statistics of the filtered datasets.

	ML1M		
	F1@5	**F1@10**	**F1@15**
Popularity	0.506	0.600	0.605
I2I-50	0.514	0.606	0.611
U2U-80	0.532	0.619	0.620
BPRMF-10	0.530	0.621	0.621
WRMF-10	***0.537***	***0.622***	***0.622***
BPRSlim	0.488	0.574	0.578
BPRMF-LOD-10	0.530	0.620	*0.622*
HolE-RT	0.514	0.603	0.604
HolE-R	0.510	0.598	0.600
HolE-T	0.516	0.603	0.605
HolE-RT-5	0.515	0.603	0.605
HolE-RT-10	0.518	0.607	0.609
HolE	0.508	0.595	0.597
HolE-PCA-RT	0.516	0.604	0.603
HolE-PCA-R	0.518	0.605	0.604
HolE-PCA-T	0.516	0.605	0.603
HolE-PCA-RT-5	0.521	0.609	0.608
HolE-PCA-RT-10	0.522	0.611	0.610
HolE-LR	0.527	0.619	**0.622**

Table 6: Results of the experimental evaluation on ML1M data. The best-performing baseline is reported in italics while the overall best configuration is highlighted in bold.

	Last.fm		
	F1@5	**F1@10**	**F1@15**
Popularity	0.568	0.541	0.398
I2I-30	0.586	0.541	0.398
U2U-30	0.590	0.541	0.398
BPRMF-10	0.584	0.541	0.398
WRMF-30	***0.591***	0.541	0.398
BPRSlim	0.582	0.541	0.398
BPRMF-LOD-10	0.588	*0.564*	*0.415*
HolE-RT	0.559	0.565	0.416
HolE-R	0.568	0.565	0.416
HolE-T	0.559	0.565	0.416
HolE-RT-5	0.564	0.565	0.416
HolE-RT-10	0.562	0.565	0.416
HolE	0.568	**0.565**	**0.416**
HolE-PCA-RT	0.559	0.565	0.416
HolE-PCA-R	0.567	0.565	0.416
HolE-PCA-T	0.559	0.565	0.416
HolE-PCA-RT-5	0.562	0.565	0.416
HolE-PCA-RT-10	0.561	0.565	0.416
HolE-LR	0.552	0.565	0.416

Table 7: Results of the experimental evaluation on Last.fm data. The best-performing baseline is reported in italics while the overall best configuration is highlighted in bold.

	LibraryThing		
	F1@5	**F1@10**	**F1@15**
Popularity	0.623	0.581	0.506
I2I-30	0.644	0.593	0.514
U2U-30	0.647	0.593	0.514
BPRMF-10	*0.651*	*0.596*	*0.515*
WRMF-30	0.649	0.594	*0.515*
BPRSlim	0.644	0.589	0.510
BPRMF-LOD-10	*0.651*	*0.596*	*0.515*
HolE-RT	0.651	0.596	0.515
HolE-R	0.637	0.586	0.509
HolE-T	0.649	0.595	0.515
HolE-RT-5	0.644	0.590	0.512
HolE-RT-10	0.647	0.592	0.513
HolE	0.636	0.586	0.509
HolE-PCA-RT	0.652	0.596	0.515
HolE-PCA-R	0.644	0.591	0.513
HolE-PCA-T	0.652	0.596	0.515
HolE-PCA-RT-5	0.650	0.593	0.514
HolE-PCA-RT-10	0.650	0.594	0.514
HolE-LR	**0.654**	**0.599**	**0.517**

Table 8: Results of the experimental evaluation on LibraryThing data. The best-performing baseline is reported in italics while the overall best configuration is highlighted in bold.