RDF Graph Embeddings for Content-based Recommender Systems

Jessica Rosati[1,2]
[1]University of Camerino – Piazza Cavour 19/f – 62032 Camerino, Italy
[2]Polytechnic University of Bari – Via Orabona, 4 – 70125 Bari, Italy
jessica.rosati@unicam.it

Petar Ristoski
Data and Web Science Group, University of Mannheim, B6, 26, 68159 Mannheim, Germany
petar.ristoski@informatik.uni-mannheim.de

Tommaso Di Noia
Polytechnic University of Bari – Via Orabona, 4 – 70125 Bari, Italy
tommaso.dinoia@poliba.it

Renato De Leone
University of Camerino – Piazza Cavour 19/f – 62032 Camerino, Italy
renato.deleone@unicam.it

Heiko Paulheim
Data and Web Science Group, University of Mannheim, B6, 26, 68159 Mannheim, Germany
heiko@informatik.uni-mannheim.de

ABSTRACT

Linked Open Data has been recognized as a useful source of background knowledge for building content-based recommender systems. Vast amount of RDF data, covering multiple domains, has been published in freely accessible datasets. In this paper, we present an approach that uses language modeling approaches for unsupervised feature extraction from sequences of words, and adapts them to RDF graphs used for building content-based recommender system. We generate sequences by leveraging local information from graph sub-structures and learn latent numerical representations of entities in RDF graphs. Our evaluation on two datasets in the domain of movies and books shows that feature vector representations of general knowledge graphs such as DBpedia and Wikidata can be effectively used in content-based recommender systems.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval

Keywords

Recommender System; Graph Embeddings; Linked Open Data

1. INTRODUCTION

One of the main limitations of traditional content-based recommendation approaches is that the information on which they rely is generally insufficient to elicit user's interests and characterize all the aspects of her interaction with the system. This is the main drawback of the approaches built on textual and keyword-based representations, which cannot capture complex relations among objects since they lack the semantics associated to their attributes. A process of "knowledge infusion" [40] and semantic analysis has been proposed to face this issue, and numerous approaches that incorporate ontological knowledge have been proposed, giving rise to the newly defined class of *semantics-aware* content-based recommender systems [6]. More recently the *Linked Open Data* (LOD) initiative [3] has opened new interesting possibilities to realize better recommendation approaches. The LOD initiative in fact gave rise to a variety of open knowledge bases freely accessible on the Web and being part of a huge decentralized knowledge base, the LOD cloud, where each piece of little knowledge is enriched by links to related data. LOD is an open, interlinked collection of datasets in machine-interpretable form, built on *World Wide Web Consortium* (W3C) standards as RDF[1], and SPARQL[2]. Currently the LOD cloud consists of about $1,000$ interlinked datasets covering multiple domains from life science to government data [39]. It has been shown that LOD is a valuable source of background knowledge for content-based recommender systems in many domains [12]. Given that the items to be recommended are linked to a LOD dataset, information from LOD can be exploited to determine which items are considered to be similar to the ones that the user has consumed in the past, allowing to discover hidden information and implicit relations between objects [26]. While LOD is rich in high quality data, it is still challenging to find effective and efficient way of exploiting the knowledge for content-based recommendations. So far, most of the pro-

CBRecSys 2016, September 16, 2016, Boston, MA, USA.

DOI: 10.1145/1235

[1]http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/, 2004.
[2]http://www.w3.org/TR/rdf-sparql-query/, 2008

posed approaches in the literature are supervised or semi-supervised, which means cannot work without human interaction.

In this work, we adapt language modeling approaches for latent representation of entities in RDF graphs. To do so, we first convert the graph into a set of sequences of entities using graph walks. In the second step, we use those sequences to train a neural language model, which estimates the likelihood of a sequence of entities appearing in the graph. Once the training is finished, each entity in the graph is represented with a vector of latent numerical values. Projecting such latent representation of entities into a lower dimensional feature space shows that semantically similar entities appear closer to each other. Such entity vectors can be directly used in a content-based recommender system.

In this work, we utilize two of the most prominent RDF knowledge graphs [29], i.e. DBpedia [18] and Wikidata [42]. DBpedia is a knowledge graph which is extracted from structured data in Wikipedia. The main source for this extraction are the key-value pairs in the Wikipedia infoboxes. Wikidata is a collaboratively edited knowledge graph, operated by the Wikimedia foundation[3] that also hosts various language editions of Wikipedia.

The rest of this paper is structured as follows. In Section 2, we give an overview of related work. In Section 3, we introduce our approach, followed by an evaluation in Section 4. We conclude with a summary and an outlook on future work.

2. RELATED WORK

It has been shown that LOD can improve recommender systems towards a better understanding and representation of user preferences, item features, and contextual signs they deal with. LOD has been used in content-based, collaborative, and hybrid techniques, in various recommendation tasks, i.e., rating prediction, top-N recommendations and improving of diversity in content-based recommendations. LOD datasets, e.g. DBpedia, have been used in content-based recommender systems in [11] and [12]. The former performs a semantic expansion of the item content based on ontological information extracted from DBpedia and LinkedMDB [16], the first open semantic web database for movies, and tries to derive implicit relations between items. The latter involves DBpedia and LinkedMDB too, but is an adaptation of the Vector Space Model to Linked Open Data: it represents the RDF graph as a 3-dimensional tensor where each slice is an ontological property (e.g. starring, director,...) and represents its adjacency matrix. It has been proven that leveraging LOD datasets is also effective for hybrid recommender systems [4], that is in those approaches that boost the collaborative information with additional knowledge, such as the item content. In [10] the authors propose *SPRank*, a hybrid recommendation algorithm that extracts semantic path-based features from DBpedia and uses them to compute top-N recommendations in a learning to rank approach and in multiple domains, movies, books and musical artists. SPRank is compared with numerous collaborative approaches based on matrix factorization [17, 34] and with other hybrid RS, such as *BPR-SSLIM* [25], and exhibits good performance especially in those contexts characterized by high sparsity, where the contribution of the content becomes essential. Another hybrid approach is proposed in [36], which builds on training individual base recommenders and using global popularity scores as generic recommenders. The results of the individual recommenders are combined using stacking regression and rank aggregation. Most of these approaches can be referred to as *top-down* approaches [6], since they rely on the integration of external knowledge and cannot work without human intervention. On the other side, *bottom-up* approaches ground on the *distributional hypothesis* [15] for language modeling, according to which the meaning of words depends on the context in which they occur, in some textual content. The resulting strategy is therefore unsupervised, requiring a corpora of textual documents for training as large as possible. Approaches based on the distributional hypothesis, referred to as *discriminative models*, behave as word embeddings techniques where each term (and document) becomes a point in the vector space. They substitute the term-document matrix typical of Vector Space Model with a term-context matrix on which they apply dimensionality reduction techniques such as Latent Semantic Indexing (LSI) [8] and the more scalable and incremental Random Indexing (RI) [38]. The latter has been involved in [22] and [23] to define the so called enhanced Vector Space Model (eVSM) for content-based RS, where user's profile is incrementally built summing the features vectors representing documents liked by the user and a negation operator is introduced to take into account also negative preferences.

Word embedding techniques are not limited to LSI and RI. The word2vec strategy has been recently presented in [19] and [20], and to the best of our knowldge, has been applied to item recommendations in a few works [21, 28]. In particular, [21] is an empirical evaluation of LSI, RI and word2vec to make content-based movie recommendation exploiting textual information from Wikipedia, while [28] deals with check-in venue (location) recommendations and adds a non-textual feature, the past check-ins of the user. They both draw the conclusion that word2vec techniques are promising for the recommendation task. Finally there is a single example of *product embedding* [14], namely *prod2vec*, which operates on the artificial graph of purchases, treating a purchase sequence as a "sentence" and products within the sequence as words.

3. APPROACH

In our approach, we adapt neural language models for RDF graph embeddings. Such approaches take advantage of the word order in text documents, explicitly modeling the assumption that closer words in the word sequence are statistically more dependent. In the case of RDF graphs, we follow the approach sketched in [37], considering entities and relations between entities instead of word sequences. Thus, in order to apply such approaches on RDF graph data, we have to transform the graph data into sequences of entities, which can be considered as sentences. After the graph is converted into a set of sequences of entities, we can train the same neural language models to represent each entity in the RDF graph as a vector of numerical values in a latent feature space. Such entity vectors can be directly used in a content-based recommender system.

3.1 RDF Graph Sub-Structures Extraction

We propose random graph walks as an approach for con-

[3]http://wikimediafoundation.org/

verting graphs into a set of sequences of entities.

DEFINITION 1. *An RDF graph is a graph G = (V, E), where V is a set of vertices, and E is a set of directed edges.*

The objective of the conversion functions is for each vertex $v \in V$ to generate a set of sequences S_v, where the first token of each sequence $s \in S_v$ is the vertex v followed by a sequence of tokens, which might be edges, vertices, or any substructure extracted from the RDF graph, in an order that reflects the relations between the vertex v and the rest of the tokens, as well as among those tokens.

In this approach, for a given graph $G = (V, E)$, for each vertex $v \in V$ we generate all graph walks P_v of depth d rooted in the vertex v. To generate the walks, we use the breadth-first algorithm. In the first iteration, the algorithm generates paths by exploring the direct outgoing edges of the root node v_r. The paths generated after the first iteration will have the following pattern v_r ->e_{1i}, where $i \in E(v_r)$. In the second iteration, for each of the previously explored edges the algorithm visits the connected vertices. The paths generated after the second iteration will follow the following pattern v_r ->e_{1i} ->v_{1i}. The algorithm continues until d iterations are reached. The final set of sequences for the given graph G is the union of the sequences of all the vertices $\bigcup_{v \in V} P_v$.

3.2 Neural Language Models – word2vec

Until recently, most of the Natural Language Processing systems and techniques treated words as atomic units, representing each word as a feature vector using a one-hot representation, where a word vector has the same length as the size of a vocabulary. In such approaches, there is no notion of semantic similarity between words. While such approaches are widely used in many tasks due to their simplicity and robustness, they suffer from several drawbacks, e.g., high dimensionality and severe data sparsity, which limit the performance of such techniques. To overcome such limitations, neural language models have been proposed, inducing low-dimensional, distributed embeddings of words by means of neural networks. The goal of such approaches is to estimate the likelihood of a specific sequence of words appearing in a corpus, explicitly modeling the assumption that closer words in the word sequence are statistically more dependent.

While some of the initially proposed approaches suffered from inefficient training of the neural network models, with the recent advancements in the field several efficient approaches has been proposed. One of the most popular and widely used is the word2vec neural language model [19, 20]. Word2vec is a particularly computationally-efficient two-layer neural net model for learning word embeddings from raw text. There are two different algorithms, the Continuous Bag-of-Words model (CBOW) and the Skip-Gram model.

3.2.1 Continuous Bag-of-Words Model

The CBOW model predicts target words from context words within a given window.The input layer is comprised from all the surrounding words for which the input vectors are retrieved from the input weight matrix, averaged, and projected in the projection layer. Then, using the weights from the output weight matrix, a score for each word in the vocabulary is computed, which is the probability of the word being a target word. Formally, given a sequence of training words $w_1, w_2, w_3, ..., w_T$, and a context window c, the objective of the CBOW model is to maximize the average log probability:

$$\frac{1}{T}\sum_{t=1}^{T} log\ p(w_t|w_{t-c}...w_{t+c}), \quad (1)$$

where the probability $p(w_t|w_{t-c}...w_{t+c})$ is calculated using the softmax function:

$$p(w_t|w_{t-c}...w_{t+c}) = \frac{exp(\bar{v}^T v'_{w_t})}{\sum_{w=1}^{V} exp(\bar{v}^T v'_w)}, \quad (2)$$

where v'_w is the output vector of the word w, V is the complete vocabulary of words, and $\bar{v}$ is the averaged input vector of all the context words:

$$\bar{v} = \frac{1}{2c}\sum_{-c \le j \le c, j \ne 0} v_{w_{t+j}} \quad (3)$$

3.2.2 Skip-Gram Model

The Skip-Gram model does the inverse of the CBOW model and tries to predict the context words from the target words. More formally, given a sequence of training words $w_1, w_2, w_3, ..., w_T$, and a context window c, the objective of the skip-gram model is to maximize the following average log probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c \le j \le c, j \ne 0} log\ p(w_{t+j}|w_t), \quad (4)$$

where the probability $p(w_{t+j}|w_t)$ is calculated using the softmax function:

$$p(w_o|w_i) = \frac{exp(v'^T_{wo} v_{wi})}{\sum_{w=1}^{V} exp(v'^T_w v_{wi})}, \quad (5)$$

where v_w and v'_w are the input and the output vector of the word w, and V is the complete vocabulary of words.

In both cases, calculating the softmax function is computationally inefficient, as the cost for computing is proportional to the size of the vocabulary. Therefore, two optimization techniques have been proposed, i.e., hierarchical softmax and negative sampling [20]. The empirical studies show that in most cases negative sampling leads to better performances than hierarchical softmax, which depends on the selected negative samples, but it has higher runtime.

Once the training is finished, semantically similar words appear close to each other in the feature space. Furthermore, basic mathematical functions can be performed on the vectors, to extract different relations between the words.

4. EVALUATION

We evaluate different variants of our approach on two distinct datasets, and compare them to common approaches for creating content-based item representations from LOD and with state of the art collaborative approaches. Furthermore, we investigate the use of two different LOD datasets as background knowledge, i.e., DBpedia and Wikidata.

4.1 Datasets

In order to test the effectiveness of our proposal, we evaluate it in terms of ranking accuracy and aggregate diversity on two datasets belonging to different domains, i.e. `Movielens` 1M[4] for movies and `LibraryThing`[5] for books. The

[4]http://grouplens.org/datasets/movielens/
[5]https://www.librarything.com/

former contains 1 million 1-5 stars ratings from 6,040 users on 3,883 movies. The `LibraryThing` dataset contains more than 2 millions ratings from 7,564 users on 39,515 books. As there are many duplicated ratings in the dataset, when a user has rated more than once the same item, we select her last rating. This choice brings to have 626,000 ratings in the range from 1 to 10. The user-item interactions contained in the datasets are enriched with side information thanks to the item mapping and linking to DBpedia technique detailed in [27], whose dump is available at http://sisinflab.poliba.it/semanticweb/lod/recsys/datasets/. In the attempt to reduce the popularity bias from our final evaluation we decided to remove the top 1% most popular items from both datasets [5]. Moreover we keep out, from `LibraryThing`, users with less than five ratings and items rated less than five times, and to have a dataset characterized by lower sparsity we retain for `Movielens` only users with at least fifty ratings, as already done in [10]. Table 1 contains the final statistics for our datasets.

	Movielens	**LibraryThing**
Number of users	4,186	7,149
Number of items	3,196	4,541
Number of ratings	822,597	352,123
Data sparsity	93.85%	98.90%

Table 1: Statistics about the two datasets

4.1.1 RDF Embeddings

As RDF datasets we use DBpedia and Wikidata.

We use the English version of the 2015-10 DBpedia dataset, which contains $4,641,890$ instances and $1,369$ mapping-based properties. In our evaluation we only consider object properties, and ignore the data properties and literals.

For the Wikidata dataset we use the simplified and derived RDF dumps from 2016-03-28[6]. The dataset contains $17,340,659$ entities in total. As for the DBpedia dataset, we only consider object properties, and ignore the data properties and literals.

4.2 Evaluation Protocol

As evaluation protocol for our comparison, we adopted the *all unrated items* methodology presented in [41] and already used in [10]. Such methodology asks to predict a score for each item not rated by a user, irrespective of the existence of an actual rating, and to compare the recommendation list with the test set.

The metrics involved in the experimental comparison are precision, recall and nDCG as accuracy metrics, and catalog coverage and Gini coefficient for the aggregate diversity. *precision@N* represents the fraction of relevant items in the top-N recommendations. *recall@N* indicates the fraction of relevant items, in the user test set, occurring in the top-N list. As relevance threshold, we set 4 for `Movielens` and 8 for `LibraryThing`, as previously done in [10]. Although precision and recall are good indicators to evaluate the accuracy of a recommendation engine, they are not rank-sensitive. *nDCG@N* [2] instead takes into account also the position in the recommendation list, being defined as

$$\text{nDCG@}N = \frac{1}{\text{iDCG}} \cdot \sum_{i=1}^{N} \frac{2^{rel(u,i)} - 1}{log_2(1+i)} \tag{6}$$

where $rel(u,i)$ is a boolean function representing the relevance of item i for user u and iDCG is a normalization factor that sets nDCG@N value to 1 when an ideal ranking is returned [2]. As suggested in [41] and set up in [10], in the computation of nDCG@N we fixed a default "neutral" value for those items with no ratings, i.e. 3 for `Movielens` and 5 for `LibraryThing`.

Providing accurate recommendations has been recognized as just one of the main task a recommender system must be able to perform. We therefore evaluate the contribution of our latent features in terms of aggregate diversity, and more specifically by means of catalog coverage and Gini coefficient [1]. The *catalog coverage* represents the percentage of available candidate items recommended at least once. It is an important quality dimension for both user and business perspective [13], since it exhibits the capacity to not settle just on a subset of items (e.g. the most popular). This metric however should be supported by a distribution metric which has to show the ability of a recommendation engine to equally spread out the recommendations across all users. *Gini coefficient* [1] is used for this purpose, since it measures the concentration degree of top-N recommendations across items and is defined as

$$\text{Gini} \quad = \quad 2\sum_{i=1}^{n} \left(\frac{n+1-i}{n+1}\right) \cdot \left(\frac{rec(i)}{total}\right) \tag{7}$$

In Equation (7), n is the number of candidate items available for recommendation, *total* represents the total number of top-N recommendations made across all users, and $rec(i)$ is the number of users to whom item i has been recommended. Gini coefficient gives therefore an idea of the "equity" in the distribution of the items. It is worth to remind that we are following the notion given in [1], where the complement of the standard Gini coefficient is used, so that higher values correspond to more balanced recommendations.

4.3 Experimental Setup

The first step of our approach is to convert the RDF graphs into a set of sequences. Therefore, to extract the entities embeddings for the large RDF datasets, we use only random graph walks entity sequences. More precisely, we follow the approach presented in [32] to generate only a limited number of random walks for each entity. For DBpedia, we experiment with 500 walks per entity with depth of 4 and 8, while for Wikidata, we use only 200 walks per entity with depth of 4. Additionally, for each entity in DBpedia and Wikidata, we include all the walks of depth 2, i.e., direct outgoing relations. We use the corpora of sequences to build both CBOW and Skip-Gram models with the following parameters: window size = 5; number of iterations = 5; negative sampling for optimization; negative samples = 25; with average input vector for CBOW. We experiment with 200 and 500 dimensions for the entities' vectors. All the models are publicly available[7].

We compare our approach to several baselines. For generating the data mining features, we use three strategies that

[6]http://tools.wmflabs.org/wikidata-exports/rdf/index.php?content=dump_download.php\&dump=20160328

[7]http://data.dws.informatik.uni-mannheim.de/rdf2vec/

take into account the direct relations to other resources in the graph [30], and two strategies for features derived from graph sub-structures [7]:

- Features derived from specific relations. In the experiments we use the relations *rdf:type* (types), and *dcterms:subject* (categories) for datasets linked to DBpedia.
- Features derived from generic relations, i.e., we generate a feature for each incoming (rel in) or outgoing relation (rel out) of an entity, ignoring the value of the relation.
- Features derived from generic relations-values, i.e, we generate feature for each incoming (rel-vals in) or outgoing relation (rel-vals out) of an entity including the value of the relation.
- Kernels that count substructures in the RDF graph around the instance node. These substructures are explicitly generated and represented as sparse feature vectors.
  - The Weisfeiler-Lehman (WL) graph kernel for RDF [7] counts full subtrees in the subgraph around the instance node. This kernel has two parameters, the subgraph depth d and the number of iterations h (which determines the depth of the subtrees). We use $d = 1$ and $h = 2$ and therefore we will indicate this strategy as WL12.
  - The Intersection Tree Path kernel for RDF [7] counts the walks in the subtree that span from the instance node. Only the walks that go through the instance node are considered. We will therefore refer to it as the root Walk Count (WC) kernel. The root WC kernel has one parameter: the length of the paths l, for which we test 2. This strategy will be denoted accordingly as WC2.

The strategies for creating propositional features from Linked Open Data are implemented in the RapidMiner LOD extension[8] [31, 35].

4.4 Results

The target of the experimental section of this paper is two-fold. On the one hand, we want to prove that the latent features we extracted are able to subsume the other kind of features in terms of accuracy and aggregate diversity. On the other hand we aim at qualifying our strategies as valuable means for the recommendation task, through a first comparison with state of the art approaches. Both goals are pursued implementing an item-based K-nearest-neighbor method, hereafter denoted as ItemKNN, with cosine similarity among features vectors. Formally, this method determines similarities between items through cosine similarity between relative vectors and then selects a subset of them – the neighbors – for each item, that will be used to estimate the rating of user u for a new item i as follows:

$$r^*(u,i) = \sum_{j \in ratedItems(u)} cosineSim(j,i) \cdot r_{u,j}$$

where $ratedItems(u)$ is the set of items already evaluated by user u, $r_{u,j}$ indicates the rating for item j by user u and $cosineSim(j,i)$ is the cosine similarity score between items j and i. In our experiments, the size of the considered neighbourhood is limited to 5. The computation of recommendations has been done with the publicly available library RankSys[9]. All the results have been computed @10, that is considering the top-10 lists recommended to the users: precision, recall and nDCG are computed for each user and then averaged across all users, while diversity metrics are global measures.

Tables 2 and 3 contain the values of precision, recall and nDCG, respectively for `Movielens` and `LibraryThing`, for each kind of features we want to test. The best approach for both datasets is retrieved with a Skip-Gram model and with a size of 200 for vectors built upon DBpedia. For the sake of truth, on the `Movielens` dataset the highest value of precision is achieved using vector size of 500, but the size 200 is prevalent according to the F1 measure, i.e. the harmonic mean of precision and recall. A substantial difference however concerns the exploratory depth of the random walks, since for `Movielens` the results related to depth 4 outdo those computed with depth 8, while the tendency is reversed for `LibraryThing`. The advantage of the Skip-Gram model over the CBOW is a constant both on DBpedia and Wikidata. Moreover, the employment of the Wikidata RDF dataset turns out to be more effective for `LibraryThing`, where the Skip-Gram vectors with depth 4 exceeds the corresponding DBpedia vectors. Moving to the features extracted from direct relations, the contribution of the "categories" stands clearly out, together with relations-values "rel-vals", especially when just incoming relations are considered. The extraction of features from graph structures, i.e. WC2 and WL12 approaches, seems not to provide significant advantages to the recommendation algorithm.

To point out that our latent features are able to capture the structure of the RDF graph, placing closely semantically similar items, we provide some examples of the neighbouring sets retrieved using our graph embeddings technique and used within the ItemKNN. Table 4 is related to movies and displays that neighboring items are highly relevant and close to the query item, i.e. the item for which neighbors are searched for.

To further analyse the semantics of the vector representations, we employ Principal Component Analysis (PCA) to project the "high"-dimensional entities' vectors in a two dimensional feature space, or 2D scatter plot. For each of the query movies in Table 4 we visualize the vectors of the 5 nearest neighbors as shown in Figure 1. The figure illustrates the ability of the model to automatically cluster the movies.

The impact on the aggregate diversity. As a further validation of the interactiveness of our latent features for recommendation task, we report the performances of the ItemKNN approach in terms of aggregate diversity. The relation between accuracy and aggregate diversity has gained the attention of researchers in the last few years and is generally characterized as a trade-off [1]. Quite surprisingly, however, the increase in accuracy, shown in Tables 2 and 3, seems not to rely on a concentration on a subset of items, e.g. the most

[8] http://dws.informatik.uni-mannheim.de/en/research/rapidminer-lod-extension

[9] http://ranksys.org/

Strategy	P@10	R@10	nDCG@10
DB2vec CBOW 200 4	0.03893	0.02167	0.30782
DB2vec CBOW 500 4	0.03663	0.02088	0.30557
DB2vec SG 200 4	0.05681	**0.03119**	**0.31828**
DB2vec SG 500 4	**0.05786**	0.0304	0.31726
DB2vec CBOW 200 8	0.01064	0.00548	0.29245
DB2vec CBOW 500 8	0.01137	0.00567	0.29289
DB2vec SG 200 8	0.04424	0.02693	0.30997
DB2vec SG 500 8	0.02191	0.01478	0.29863
WD2vec CBOW 200 4	0.01217	0.00596	0.29362
WD2vec CBOW 500 4	0.01027	0.00427	0.29211
WD2vec SG 200 4	0.02902	0.01479	0.30189
WD2vec SG 500 4	0.02644	0.01246	0.29967
types	0.00313	0.00145	0.28864
categories	0.0305	0.02093	0.30444
rel in	0.01122	0.00589	0.29183
rel out	0.02844	0.01607	0.30274
rel in & out	0.02852	0.01566	0.3006
rel-vals in	0.03883	0.02293	0.29411
rel-vals out	0.01279	0.00971	0.29378
rel-vals in & out	0.01174	0.00913	0.29333
WC2	0.00684	0.00343	0.29032
WL12	0.00601	0.00288	0.28977

Table 2: Results of the ItemKNN approach on `Movielens` dataset. P and R stand respectively for precision and recall, SG indicates the Skip-Gram model, and DB and WD represent DBpedia and Wikidata respectively.

Strategy	P@10	R@10	nDCG@10
DB2vec CBOW 200 4	0.05127	0.11777	0.21244
DB2vec CBOW 500 4	0.05065	0.11557	0.21039
DB2vec SG 200 4	0.05719	0.12763	0.2205
DB2vec SG 500 4	0.05811	0.12864	0.22116
DB2vec CBOW 200 8	0.00836	0.02334	0.14147
DB2vec CBOW 500 8	0.00813	0.02335	0.14257
DB2vec SG 200 8	**0.07681**	**0.17769**	**0.25234**
DB2vec SG 500 8	0.07446	0.1743	0.24809
WD2vec CBOW 200 4	0.00537	0.01084	0.13524
WD2vec CBOW 500 4	0.00444	0.00984	0.13428
WD2vec SG 200 4	0.06416	0.14565	0.23309
WD2vec SG 500 4	0.06031	0.14194	0.22752
types	0.01854	0.04535	0.16064
categories	0.06662	0.15258	0.23733
rel in	0.04577	0.10219	0.20196
rel out	0.04118	0.09055	0.19449
rel in & out	0.04531	0.10165	0.20115
rel-vals in	0.06176	0.14101	0.22574
rel-vals out	0.06163	0.13763	0.22826
rel-vals in & out	0.06087	0.13662	0.22615
WC2	0.00159	0.00306	0.12858
WL12	0.00155	0.00389	0.12937

Table 3: Results of the ItemKNN approach on `LibraryThing` dataset.

popular ones, according to the results proposed in Tables 5 and 6. Here we are reporting, for the sake of concisenesses, only the best approaches for each kind of features. More clearly, we are displaying the best approach for latent features computed on DBpedia, the best approach for latent features computed on Wikidata and the values for the strategy involving categories, since it provides the highest scores among features extracted through direct relations. We are not reporting the values related to WL12 and WC2 algorithms, since their contribution is rather low also in this

Query Movie	K Nearest Neighbours
Batman	Batman Forever, Batman Returns, Batman & Robin, Superman IV: The Quest for Peace, Dick Tracy
Bambi	Cinderella, Dumbo, 101 Dalmatians , Pinocchio, Lady and the Tramp
Star Trek: Generations	Star Trek VI: The Undiscovered Country, Star Trek: Insurrection, Star Trek III: The Search for Spock, Star Trek V: The Final Frontier, Star Trek: First Contact (1996)

Table 4: Examples of K-nearest-neighbor sets on `Movielens`, for the Skip-Gram model with depth of 4 and size vectors 200, on DBpedia.

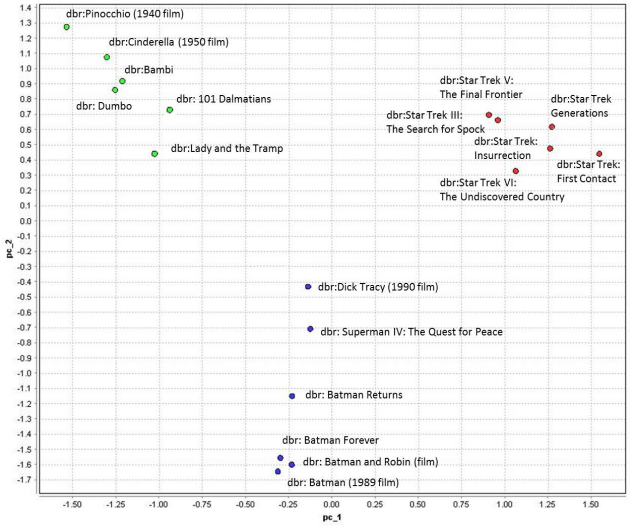

Figure 1: Two-dimensional PCA projection of the 200-dimensional Skip-gram vectors of movies in Table 4.

analysis. For both movies and books domain, the best approaches found on DBpedia for the accuracy metrics, i.e. respectively "DB2vec SG 200 4" and "DB2vec SG 200 8", perform better also in terms of aggregate diversity. For the `LibraryThing` dataset the Skip-Gram model computed with random walks on Wikidata and size vector limited to 200 is very close to the highest scores retrieved in DBpedia, while for `Movielens` is the CBOW model, with depth 4, to gain the best performance on Wikidata. The contribution of the categories, despite being lower than the best approach on each dataset, is quite significant for diversity measures too.

Comparison with state of the art collaborative approaches. It is a quite common belief in the RS field that using pure content-based approaches would not be enough to provide accurate suggestions and that the recommendation engines must ground on collaborative information too. This motivated us to explicitly compare the best approaches built on graph embeddings technique with the well-known state of the art collaborative recommendation algorithms listed be-

Strategy	Coverage	Gini
DB2vec SG 200 4	**0.35198**	**0.07133**
WD2vec CBOW 200 4	0.27749	0.04052
categories	0.29798	0.04714

Table 5: Methods comparison in terms of aggregate diversity on the `Movielens` dataset. Coverage stands for catalog coverage and Gini for Gini coefficient.

Strategy	Coverage	Gini
DB2vec SG 200 8	**0.76386**	**0.29534**
WD2vec SG 200 4	0.73037	0.28525
categories	0.7246	0.26409

Table 6: Methods comparison in terms of aggregate diversity on the `LibraryThing` dataset.

low, and implemented with the publicly available software library MyMediaLite[10].

- Biased Matrix Factorization (MF) [17], recognized as the state of the art for rating prediction, is a matrix factorization model that minimizes RMSE using stochastic gradient descent and both user and item bias.
- PopRank is a baseline based on popularity. It recommends the same recommendations to all users according to the overall items popularity. Recent studies have point out that recommending the most popular items could already result in a high performance [5].
- Bayesian Personalized Ranking (BPRMF) combines a matrix factorization approach with a Bayesian Personalized Ranking optimization criterion [34].
- SLIM [24] is a Sparse LInear Method for top-N recommendation that learns a sparse coefficient matrix for the items involved in the system by only relying on the users purchase/ratings profile and by solving a L1-norm and L2-norm regularized optimization problem.
- Soft Margin Ranking Matrix Factorization (RankMF) is a matrix factorization approach for ranking, whose loss function is ordinal regression [43].

Tables 7 and 8 provide the comparison results for `Movielens` and `LibraryThing` respectively. Table 7 shows that matrix factorization techniques and the SLIM algorithm exceed our approach based only on content information. This outcome was somehow expected, especially considering that, in our experimental setting, `Movielens` dataset retains only users with at least fifty ratings. The community-based information is unquestionably predominant for this dataset, whose sparsity would probably be unlikely for most real-world scenarios. The behaviour however is completely overturned on the `LibraryThing` dataset, whose results are collected in Table 8. In this case, the mere use of our features vectors (i.e. the "DB2vec SG 200 8" strategy) is able to outperform the competitor algorithms, which are generally regarded as the most efficient collaborative algorithms for both rating and ranking prediction.

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

Strategy	P@10	R@10	nDCG@10
DB2vec SG 200 4	0.0568	0.0312	0.3183
MF	0.2522	0.1307	0.4427
PopRank	0.1673	0.0787	0.3910
BPRMF	0.2522	0.1307	0.4427
SLIM	**0.2632**	**0.1474**	**0.4599**
RankMF	0.1417	0.0704	0.3736

Table 7: Comparison with state of the art collaborative approaches on `Movielens`.

Strategy	P@10	R@10	nDCG@10
DB2vec SG 200 8	**0.0768**	**0.1777**	**0.2523**
MF	0.0173	0.0209	0.1423
PopRank	0.0397	0.0452	0.1598
BPRMF	0.0449	0.0751	0.1858
SLIM	0.0543	0.0988	0.2317
RankMF	0.0369	0.0459	0.1714

Table 8: Comparison with state of the art collaborative approaches on `LibraryThing`.

5. CONCLUSION

In this paper, we have presented an approach for learning low-dimensional real-valued representations of entities in RDF graphs, in a completely domain independent way. We have first converted the RDF graphs into a set of sequences using graph walks, which are then used to train neural language models. In the experimental section we have shown that a content-based RS relying on the similarity between items computed according to our latent features vectors, outdo the same kind of system but grounding on explicit features (e.g. types, categories,...) or features generated with the use of kernels, from both perspectives of accuracy and aggregate diversity. Our purely content-based system has been further compared to state of the arts collaborative approaches for rating prediction and item ranking, giving outstanding results on a dataset with a realistic sparsity degree.

As future work, we intend to introduce the features vectors deriving from the graph embeddings technique within a hybrid recommender system in order to get a fair comparison against state of the art hybrids approaches such as SPRank [10] and BRP-SSLIM [25]. In this perspective we could take advantage of the Factorization Machines [33], general predictor working with any features vector, that combine Support Vector Machines and factorization models. We aim to extend the evaluation to additional metrics, such as the individual diversity [44, 9], and to provide a deeper insight into cold-start users, i.e. users with a small interaction with the system for whom the information inference is difficult to draw and that generally benefit most of content "infusion".

6. REFERENCES

[1] Gediminas Adomavicius and YoungOk Kwon. Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Trans. on Knowl. and Data Eng.*, 24(5):896–911, May 2012.

[2] Alejandro Bellogín, Iván Cantador, and Pablo Castells. A comparative study of heterogeneous item recommendations in social systems. *Inf. Sci.*, 221:142–169, February 2013.

[3] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked Data – The Story So Far. *International journal on semantic web and information systems*, 5(3):1–22, 2009.

[4] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, November 2002.

[5] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, pages 39–46, New York, NY, USA, 2010. ACM.

[6] Marco de Gemmis, Pasquale Lops, Cataldo Musto, Narducci Fedelucio, and Giovanni Semeraro. Semantics-aware content-based recommender systems. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 119–159. Springer, 2nd edition, 2015.

[7] Gerben Klaas Dirk de Vries and Steven de Rooij. Substructure counting graph kernels for machine learning from rdf data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 35:71–84, 2015.

[8] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE*, 41(6):391–407, 1990.

[9] T. Di Noia, V. C. Ostuni, J. Rosati, P. Tomeo, and E. Di Sciascio. An analysis of users' propensity toward diversity in recommendations. In *ACM RecSys '14*, RecSys '14, pages 285–288. ACM, 2014.

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

[11] Tommaso Di Noia, Roberto Mirizzi, Vito Claudio Ostuni, and Davide Romito. Exploiting the web of data in model-based recommender systems. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, pages 253–256, New York, NY, USA, 2012. ACM.

[12] Tommaso Di Noia, Roberto Mirizzi, Vito Claudio Ostuni, Davide Romito, and Markus Zanker. Linked open data to support content-based recommender systems. In *Proceedings of the 8th International Conference on Semantic Systems*, I-SEMANTICS '12, pages 1–8, New York, NY, USA, 2012. ACM.

[13] Mouzhi Ge, Carla Delgado-battenfeld, and Dietmar Jannach. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *In RecSys '10*, page 257, 2010.

[14] Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp. E-commerce in your inbox: Product recommendations at scale. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1809–1818, New York, NY, USA, 2015. ACM.

[15] Z. S. Harris. *Mathematical Structures of Language*. Wiley, New York, NY, USA, 1968.

[16] Oktie Hassanzadeh and Mariano Consens. M.: Linked movie data base. In *In: Workshop on Linked Data on the Web*, 2009.

[17] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

[18] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, SÃűren Auer, and Christian Bizer. DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal*, 2013.

[19] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[20] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[21] Cataldo Musto, Giovanni Semeraro, Marco De Gemmis, and Pasquale Lops. Word embedding techniques for content-based recommender systems: an empirical evaluation. In *RecSys Posters, ser. CEUR Workshop Proceedings, P. Castells, Ed*, volume 1441.

[22] Cataldo Musto, Giovanni Semeraro, Pasquale Lops, and Marco de Gemmis. *Random Indexing and Negative User Preferences for Enhancing Content-Based Recommender Systems*, pages 270–281. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[23] Cataldo Musto, Giovanni Semeraro, Pasquale Lops, and Marco de Gemmis. *Contextual eVSM: A Content-Based Context-Aware Recommendation Framework Based on Distributional Semantics*, pages 125–136. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[24] Xia Ning and George Karypis. SLIM: sparse linear methods for top-n recommender systems. In *11th IEEE International Conference on Data Mining, ICDM 2011, Vancouver, BC, Canada, December 11-14, 2011*, pages 497–506, 2011.

[25] Xia Ning and George Karypis. Sparse linear methods with side information for top-n recommendations. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, pages 155–162, New York, NY, USA, 2012. ACM.

[26] Tommaso Di Noia, Vito Claudio Ostuni, Jessica Rosati, Paolo Tomeo, Eugenio Di Sciascio, Roberto Mirizzi, and Claudio Bartolini. Building a relatedness graph from linked open data: A case study in the it domain. *Expert Systems with Applications*, 44:354 – 366, 2016.

[27] V. C. Ostuni, T. Di Noia, E. Di Sciascio, and R. Mirizzi. Top-n recommendations from implicit feedback leveraging linked open data. In *ACM RecSys '13*, pages 85–92, 2013.

[28] Makbule Gulcin Ozsoy. From word embeddings to item recommendation. *arXiv preprint arXiv:1601.01356*, 2016.

[29] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, (Preprint):1–20, 2016.

[30] Heiko Paulheim and Johannes Fümkranz. Unsupervised generation of data mining features from linked open data. In *Proceedings of the 2nd international conference on web intelligence, mining and semantics*, page 31. ACM, 2012.

[31] Heiko Paulheim, Petar Ristoski, Evgeny Mitichkin, and Christian Bizer. Data mining with background knowledge from the web. *RapidMiner World*, 2014.

[32] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

[33] Steffen Rendle. Factorization machines with libfm. *ACM Trans. Intell. Syst. Technol.*, 3(3):57:1–57:22, May 2012.

[34] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, pages 452–461, Arlington, Virginia, United States, 2009.

[35] Petar Ristoski, Christian Bizer, and Heiko Paulheim. Mining the web of linked data with rapidminer. *Web Semantics: Science, Services and Agents on the World Wide Web*, 35:142–151, 2015.

[36] Petar Ristoski, Eneldo Loza Mencía, and Heiko Paulheim. A hybrid multi-strategy recommender system using linked open data. In *Semantic Web Evaluation Challenge*, pages 150–156. Springer, 2014.

[37] Petar Ristoski and Heiko Paulheim. Rdf2vec: Rdf graph embeddings for data mining. In *International Semantic Web Conference (To Appear)*. Springer, 2016.

[38] Magnus Sahlgren. An introduction to random indexing. In *In Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE 2005*, 2005.

[39] Max Schmachtenberg, Christian Bizer, and Heiko Paulheim. Adoption of the linked data best practices in different topical domains. In *International Semantic Web Conference*, pages 245–260. Springer, 2014.

[40] Giovanni Semeraro, Pasquale Lops, Pierpaolo Basile, and Marco de Gemmis. Knowledge infusion into content-based recommender systems. In *Proceedings of the Third ACM Conference on Recommender Systems*, RecSys '09, pages 301–304, New York, NY, USA, 2009. ACM.

[41] Harald Steck. Evaluation of recommendations: Rating-prediction and ranking. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, pages 213–220, New York, NY, USA, 2013. ACM.

[42] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[43] Markus Weimer, Alexandros Karatzoglou, and Alex Smola. Improving maximum margin matrix factorization. *Mach. Learn.*, 72(3):263–276, September 2008.

[44] Mi Zhang and Neil Hurley. Avoiding monotony: Improving the diversity of recommendation lists. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 123–130, New York, NY, USA, 2008. ACM.