Automatic Selection of Linked Open Data features in Graph-based Recommender Systems

Cataldo Musto, Pierpaolo Basile, Marco de Gemmis,
Pasquale Lops, Giovanni Semeraro, Simone Rutigliano
Università degli Studi di Bari "Aldo Moro" - Italy
name.surname@uniba.it

ABSTRACT

In this paper we compare several techniques to automatically feed a graph-based recommender system with features extracted from the Linked Open Data (LOD) cloud. Specifically, we investigated whether the integration of LOD-based features can improve the effectiveness of a graph-based recommender system and to what extent the choice of the features selection technique can influence the behavior of the algorithm by *endogenously* inducing a higher *accuracy* or a higher *diversity*. The experimental evaluation showed a clear correlation between the choice of the feature selection technique and the ability of the algorithm to maximize a specific evaluation metric. Moreover, our algorithm fed with LOD-based features was able to overcome several state-of-the-art baselines: this confirmed the effectiveness of our approach and suggested to further investigate this research line.

Keywords

Recommender Systems, PageRank, Graphs, Linked Open Data, Feature Selection, Diversity

1. BACKGROUND

The Linked Open Data (LOD) cloud is a huge set of interconnected RDF statements covering many topical domains, ranging from government and geographical data to structured information about media (movies, books, etc.) and life sciences. The typical *entry point* to all this plethora of data is DBpedia [1], the RDF mapping of Wikipedia, which is commonly considered as the *nucleus* of the emerging *Web of Data*. Thanks to the wide-spread availability of this free machine-readable knowledge, a big effort is now spent to investigate whether and how the data gathered from the LOD cloud can be exploited to improve intelligent and adaptive applications, such a Recommender System (RS).

Recent attempts towards the exploitation of Linked Open Data to build RSs are due to Passant [6], who proposed a music recommender system based on semantic similarity calculations based on DBpedia properties. The use of DBpedia for similarity calculation is also the core of the work presented by Musto et al. in [4]: in this paper user preferences in music extracted from Facebook are used as input to find other relevant artists and to build a personalized music playlist. Recently, the use of LOD-based data sources has been the core of the ESWC 2014 Recommender Systems Challenge[1]: in that setting, the best-performing approaches [2] were based on ensembles of several widespread algorithms running on diverse sets of features gathered from the LOD cloud. However, none of the above described work tackles nor the issue of *automatically* selecting the best subset of LOD-based features, neither analyzes the impact of such selection techniques on different metrics as the *diversity* of the recommendations.

To this end, in this paper we propose a methodology to automatically feed a graph-based recommendation algorithm with features extracted from the LOD cloud. We focused our attention on graph-based approaches since they use a *uniform* formalism to represent both *collaborative* features (connections between users and items, expressed through ratings) and *LOD-based* ones (connections between different items, expressed through RDF statements). As graph-based algorithm we adopted PageRank with Priors [3]. Moreover, in this work we compared several techniques to automatically select the best subset of LOD-based features, with the aim to investigate to what extent the choice of the feature selection technique can influence the behavior of the algorithm and can endogenously lead to a higher *accuracy* or a higher *diversity* of the recommendations.

The rest of the paper is organized as follows: the description of our recommendation methodology is the core of Section 2, while the details of the experimental evaluation we carried out along with the discussion of the results are provided in Section 3. Finally, Section 4 sketches conclusions and future work.

2. METHODOLOGY

The main idea behind our graph-based model is to represent *users* and *items* as *nodes* in a graph. Formally, given a set of users $U = \{u_1 \dots u_n\}$ and a set of items $I = \{i_1 \dots u_m\}$, a graph $G = \langle V, E \rangle$ is instantiated. Given that for each user and for each item a node is created, $|V| = |U| + |I|$. Next, an edge connecting a user u_i with an item i_j is created for each positive feedback expressed by that user, so $E = \{(u_i, i_j) | likes(u_i, i_j) = true\}$.

CBRecSys 2015, September 20, 2015, Vienna, Austria.

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

Given this basic formulation, built on the ground of simple *collaborative*[2] data points, each item $i \in I$ can be provided with a relevance score. To calculate the relevance of each item, we used a well-known variant of the PageRank called *PageRank with Priors* [3]. Differently from PageRank, which assigns an evenly distributed prior probability to each node ($\frac{1}{N}$, where N is the number of nodes), *PageRank with Priors* adopts a non-uniform personalization vector assigning different weights to different nodes to get a bias towards some nodes (specifically, the preferences of a specific user). In our algorithm the probability was distributed by defining a simple heuristics, set after a rough tuning: 80% of the total weight is evenly distributed among items liked by the users (0% assigned to items disliked by the users), while 20% is evenly distributed among the remaining nodes. Damping factor was set equal to 0.85, as in [5].

Given this setting, the PageRank with Priors is executed for each user (this is mandatory, since the prior probabilities change according to user's feedbacks), and nodes are ranked according to their PageRank score which is in turn calculated on the ground of the connectivity in the graph. The output of the PageRank is a list of nodes ranked according to PageRank scores, labeled as L. Given L, recommendations are built by extracting from L only those nodes $i_1 \dots i_n \in I$.

2.1 Introducing LOD-based features

As stated above, our basic formulation does not take into account any data point different from users' ratings. The insight behind this work is to enrich the above described graph by introducing some *extra* nodes and some *extra* edges, defined on the ground of the information available in the LOD cloud. Formally, we want to define an extended graph $G_{LOD} = \langle V_{LOD-ALL}, E_{LOD-ALL} \rangle$, where $V_{LOD-ALL} = V \cup V_{LOD}$ and $E_{LOD-ALL} = E \cup E_{LOD}$. V_{LOD} and E_{LOD} represent the extra nodes and the extra edges instantiated by analyzing the data gathered from the LOD cloud, respectively.

As an example, if we consider the movie *The Matrix*, the property HTTP://DBPEDIA.ORG/PROPERTY/DIRECTOR encoding the information about the director of the movie is available in the LOD cloud. Consequently, an extra node *The Wachowski Brothers* is added in V_{LOD} and an extra edge, labeled with the name of the property, is instantiated in E_{LOD} to connect the movie with its director. Similarly, if we consider the property HTTP://DBPEDIA.ORG/PROPERTY/STARRING, new nodes and new edges are defined, in order to model the relationship between *The Matrix* and the main actors, as *Keanu Reeves*, for example. In turn, given that *Keanu Reeves* acted in several movies, many new edges are added in the graph and many new paths now connect different movies: these paths would not have been available if the only *collaborative data points* were instantiated.

It immediately emerges that, due to this novel enriched representation, the structure of the graph tremendously changes since many new nodes and many new edges are added to the model: the first goal of our experimental session will be to investigate whether graph-based RSs can benefit of the introduction of novel LOD-based features.

2.2 Selecting LOD-based features

Thanks to the data points available in the LOD cloud, many new information can be encoded in our graph. However, as the number of extra nodes and extra edges grows, the computational load of the PageRank with Priors grows as well, so it necessary to identify the subset of the most useful properties gathered from the LOD cloud and to investigate to what extent (if any) each of them improves the accuracy of our recommendation strategy.

A very *naive* approach may be to manually select the most relevant LOD-based features, according to simple heuristics or to domain knowledge (e.g. properties as *director*, *starring*, *composer* may be considered as relevant for the Movie domain, whereas properties as *runtime* or *country* may be not). This basic approach has several drawbacks, since it requires a manual effort, but it is also strictly domain-dependent.

To avoid this, we employed *features selection techniques* to automatically select the most promising LOD-based features. Formally, our idea is to take as input E_{LOD}, the overall set of LOD-based properties, and to produce as output $E_{LOD-FS_T} \subseteq E_{LOD}$, the set of properties a specific feature selection technique T returned as relevant. Clearly, the exploitation of a feature selection technique T also produces a set $V_{LOD-FS_T} \subseteq V_{LOD}$, containing all the LOD-based nodes connected to the properties in E_{LOD-FS_T}.

In this setting, given a FS technique T, PageRank will be executed against the graph $G_{LOD-T} = \langle V_{LOD-T}, E_{LOD-T} \rangle$, where $V_{LOD-T} = V \cup V_{LOD-FS_T}$ and $E_{LOD-T} = E \cup E_{LOD-FS_T}$. In the experimental session the effectiveness of seven different techniques for automatic selection of LOD-based features: *PageRank*, *Principal Component Analysis (PCA)*, *Support Vector Machines (SVM)*, *Chi-Squared Test (CHI)*, *Information Gain (IG)*, *Information Gain Ratio (GR)* and *Mininum Redundancy Maximum Relevance (MRMR)* [7]. Clearly, a complete description of these techniques is out of the scope of this paper. We will just limit to evaluate their impact on the overall *accuracy* and the overall *diversity* obtained by our algorithm.

3. EXPERIMENTAL EVALUATION

Our experiments were designed on the ground of four different research questions:

1. Do graph-based recommender systems benefit of the introduction of LOD-based features?
2. Do graph-based recommender systems exploiting LOD features benefit of the adoption of FS techniques?
3. Is there any correlation between the choice of the FS technique and the behavior of the algorithm?
4. How does our methodology perform with respect to state-of-the-art techniques?

Experimental design: experiments were performed by exploiting MovieLens[3] dataset, consisting of 100,000 ratings provided by 943 users on 1,682 movies. The dataset is positively balanced (55.17% of positive ratings) and shows an high sparsity (93.69%). Each user voted 84.83 items on average and each item was voted by 48.48 users, on average.

Experiments were performed by carrying out a 5-folds cross validation. Given that MovieLens preferences are expressed on a 5-point discrete scale, we decided to consider as *positive* ratings only those equal to 4 and 5. As recommendation algorithms we used the previously described PageRank

[2] We just modeled user-items couples, as in collaborative filtering algorithms

[3] http://grouplens.org/datasets/movielens/

with Priors, set as explained in Section 2. We compared the effectiveness of our graph-based recommendation methodology by considering three different graph topologies: G, modeling the basics *collaborative* information about user ratings; G_{LOD}, which enrichs G by introducing LOD-based features gathered from DBpedia, and G_{LOD-T} which lighten the load of PageRank with Priors by relying on the features selected by a FS technique T. In order to enrich the graph G, each item in the dataset was mapped to a DBpedia entry. In our experiments 1,600 MovieLens entries (95.06% of the movies) were successfully mapped to a DBpedia node. The items for which a DBpedia entry was not found were only represented by using *collaborative* data points. Overall, MovieLens entries were described through 60 different DBpedia properties. As feature selection techniques all the approaches previously mentioned were employed, while for the parameter K (the number of LOD-based features) three different values were compared: 10, 30 and 50. The performance of each graph topology was evaluated in terms of *F1-measure*. Moreover, we also calculated the overall running time[4] of each experiment. To answer the third research questions we also evaluated the *diversity* of the recommendations, calculated by exploiting the classical Intra-List Diversity (ILD). Statistical significance was assessed by exploiting Wilcoxon and Friedman tests.

Discussion of the Results: in the first experiment we evaluated the introduction of LOD-based features in graph-based recommender systems. Results are depicted in the first two columns of Table 1. As regards MovieLens, a statistically significant improvement ($p << 0.0001$, assessed through a Wilcoxon test) was obtained for all the metrics. As expected, the expansion of the graph caused an exponential growth of the run time of the algorithm. This is due to the fact that the expansion stage introduced many new nodes and many new edges in the graph (see Table 1). The growth is particularly significant since 50,000 new nodes and 78,000 new edges were added to the graph.

Next, we evaluated the impact of all the previously presented feature selection techniques in such recommendation setting. By analyzing the results provided in Table 2, it emerged that our graph-based recommendation strategy does not often benefit of the application of FS techniques. Indeed, when a very small number of properties is chosen (*K=10*), none of the configurations is able to overcome the baseline. By slightly increasing the value of parameter K (*K=30*), only three out of seven techniques (PageRank, PCA and mRMR) improve the F1-measure. Next, when more data points are introduced (with *K=50*) better results are obtained and the F1-measure of the baseline is always overcame. Given that the overall number of LOD-based properties was equal to 60, it is possible to state that most of the properties encoded in the extended graph G_{LOD} can be considered as relevant. Clearly, this is a very domain-specific outcome, which needs to be confirmed by more thorough analysis on different datasets. However, it is possible to state that the adoption of FS techniques requires a complete analysis of the usefulness of each of the properties encoded in the LOD. Overall, the best performing configuration was PCA, which was the only technique always overcoming the baseline with $K = 50$. A Friedman test also showed that PCA statistically overcomes the other techniques for all the metrics. Another interesting outcome which follows the use of FS techniques is the saving of computational resources to run PageRank with Priors on graph $G_{LOD-PCA}$. As shown in Table 1, the adoption of FS caused a huge decrease of the run time of the algorithms equal to 33.9% for MovieLens (from 880 to 581 minutes). This is due to the smaller number of information which are modeled in the graphs (-8.6% nodes and -4.8% edges).

	MovieLens		
	G	G_{LOD}	$G_{LOD+PCA}$
F1@5	0.5406	**0.5424**	**0.5424(*)**
F1@10	0.6068	**0.6083**	**0.6088(*)**
F1@15	0.5956	**0.5963**	**0.5970(*)**
F1@20	0.5678	**0.5686**	**0.5689(*)**
Run (min.)	72	880	581
K (LOD prop.)	0	60	50
Nodes	2,625	53,794	49,158
Edges	100,000	178,020	169,405

Table 1: Overall comparison among the baseline, the complete LOD-based graph and the LOD-based graph boosted by PCA. The configurations overcoming the baseline were highlighted in bold. The best-performing configuration is further highlighted with (*)

.

In *Experiment 3* we shifted the attention from F1-measure to different evaluation metrics, and we investigate whether the adoption of a specific FS technique can *endogenously* induce a higher diversity at the expense of a little F1. Results of the experiments are provided in Figure 1a. Due to space reasons, only the results for F1@10 are provided. In both charts we used four different symbols to identify the different behaviors of each technique. It emerged that CHI was the less useful technique, since it did not provide any significant benefit to neither F1 nor diversity. Next, PageRank provided a (small) improvement on F1 and it did not significantly change the diversity of the recommendations. It is noteworth that the larger increase in accuracy of PCA is balanced by a decrease in terms of diversity. On the other side, Gain Ratio obtained the overall best diversity of the recommendation but it decreases the F1 of the algorithm. To sum up, these results show that the choice of a particular FS technique has a significant impact on the overall behavior of the recommendation algorithm. As shown in the experiment, some techniques have the ability of inducing a higher diversity (or F1) at the expense of a little of F1 (or diversity, respectively), wheres other can provide a good compromise between both metrics. Clearly, more investigation is needed to deeply analyze the behavior of each technique, but these results already give some general guidelines which can drive the choice of the FS technique which best fits the requirements of a specific recommendation scenario.

Finally, we compared the effectiveness of our methodology to the current state of the art. As baselines User-to-User CF (U2U-KNN), Item-to-Item CF (I2I-KNN), a simple popularity-based approach, a random baseline and the Bayesian Personalized Ranking Matrix Factorization (BPRMF) were used. We adopted the implementations available in MyMediaLite Recommender System library[5]. As regards

[4] Experiments were run on an Intel-i7-3770 CPU3.40 gHZ, with 32GB RAM.

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

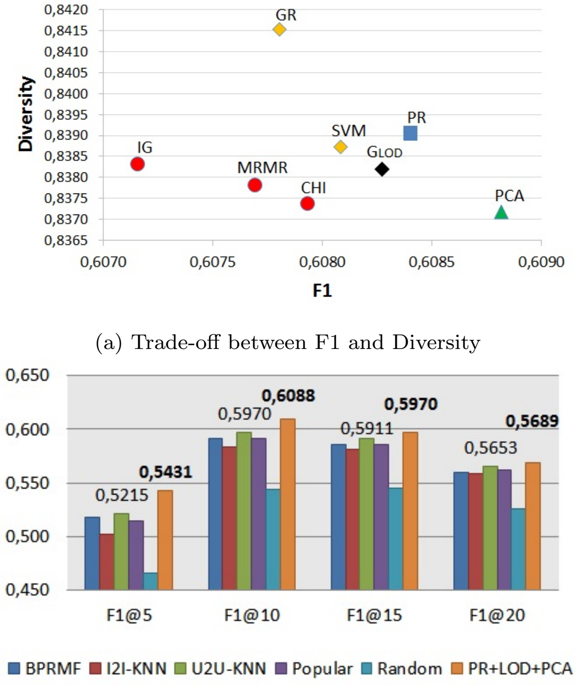
MovieLens	#feat.	**PR**	**PCA**	**SVM**	**CHI**	**IG**	**GR**	**mRMR**
F1@5	10	0.5418	0.5406	0.5382	0.5414	0.5397	0.5372	0.5397
$G_{LOD} = 0.5424$	30	**0.5429(∗)**	0.5413	0.5413	0.5419	0.5396	0.5398	**0.5429(∗)**
	50	0.5412	**0.5431(∗)(↑)**	0.5421(∗)	0.5420(∗)	0.5412(∗)	0.5406(∗)	0.5421
F1@10	10	0.6069	0.6045	0.6043	0.6056	0.6039	0.6033	0.6039
$G_{LOD} = 0.6083$	30	**0.6084(∗)**	0.6081	0.6074	0.6070	0.6055	0.6059	0.6072(∗)
	50	0.6070	**0.6088(∗)(↑)**	0.6081(∗)	0.6079(∗)	0.6072(∗)	0.6078(∗)	0.6077
F1@15	10	**0.5964**	0.5948	0.5943	0.5955	0.5950	0.5938	0.5950
$G_{LOD} = 0.5963$	30	**0.5967(∗)**	**0.5967**	**0.5964**	**0.5967**	0.5955	0.5960	0.5961
	50	0.5955	**0.5970(∗)(↑)**	**0.5966(∗)**	**0.5972(∗)**	0.5962(∗)	**0.5968(∗)**	0.5962(∗)
F1@20	10	0.5684(∗)	0.5667	0.5666	0.5672	0.5668	0.5666	0.5668
$G_{LOD} = 0.5686$	30	0.5684	**0.5688**	0.5679	0.5679	0.5675	0.5675	0.5679
	50	0.5682	**0.5689(∗)(↑)**	0.5683(∗)	**0.5686(∗)**	0.5685(∗)	**0.5687(∗)**	0.5685(∗)

Table 2: Experiment 2. The configurations overcoming the baseline G_{LOD} are emphasized in bold. Next, for each technique, the number of features which led to the highest F1 is indicated with (∗). The overall highest F1 score for each metric is highlighted with (∗)(↑). The column of the feature selection technique which performed the best on a specific dataset is coloured in grey.

U2U and I2I, neighborhood size was set to 80, while BPRMF was run by setting the factor parameter equal to 100. Results are depicted in Figure 1b. As shown in the plots, our graph-based RS outperforms all the baselines for all the metrics taken into account. It is worth to note that our approach obtained a higher F1 also when compared to a well-perfoming matrix factorization algorithm as BPRMF, thus this definitely confirmed the effectiveness of our approach.

4. CONCLUSIONS AND FUTURE WORK

In this work we proposed a graph-based recommendation methodology based on PageRank with Priors, and we evaluated different techniques to automatically feed such a representation with features extracted from the LOD cloud. Results showed that graph-based RSs can benefit of the infusion of novel knowledge coming from the LOD cloud and that a clear correlation between the adoption of a specific FS technique with the overall results of the recommender exitss, since some techniques *endogenously* showed the ability of increasing also the diversity of the recommendations generated by the algorithm. We also showed that our methodology was able to overcome several state-of-the-art baselines on both datasets. As future work, we will validate the approach by evaluating it on different dataset, and we will investigate the impact of LOD-based features with different learning approaches as Random Forest or SVM.



(a) Trade-off between F1 and Diversity

(b) Comparisons to baselines

Figure 1: Results of the experiments

5. REFERENCES

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. *DBpedia: A nucleus for a Web of Open Data*. Springer, 2007.

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

[3] T. H. Haveliwala. Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search. *IEEE Trans. Knowl. Data Eng.*, 15(4):784–796, 2003.

[4] C. Musto, G. Semeraro, P. Lops, M. de Gemmis, and F. Narducci. Leveraging social media sources to generate personalized music playlists. In *EC-Web 2012*, volume 123 of *LNBIP*, pages 112–123. Springer, 2012.

[5] L. Page, S. Brin, R. Motwani, and T. Winograd. The pageRank citation ranking: bringing order to the web. 1999.

[6] A. Passant. dbrec - Music Recommendations Using DBpedia. In *International Semantic Web Conference, Revised Papers*, volume 6497 of *LNCS*, pages 209–224. Springer, 2010.

[7] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(8):1226–1238, 2005.