A Recommendation Method Based on Personalized Random Walks

Cecília Raposo
cecilia.raposo@ist.utl.pt

Instituto Superior Técnico, INESC-ID
Av. Professor Cavaco Silva, 2744-016, Porto Salvo, Portugal

Abstract. Recommender systems analyze user preferences with the purpose of recommending items to users that meet their needs and preferences. In the context of my MSc thesis, I studied a recommendation method based on personalized random walks over a graph that encodes associations between users, items and contextual attributes. This method was tested in the setting of recommending locations with basis on data from location-based social networks such as Yelp. The graph-based representation of the problem is constructed using the explicit feedback contained in a dataset collected from Yelp. An adaptation of the Personalized PageRank algorithm was used for simulating the random walks, in order to recommend the top$-k$ local business to users. This paper describes the most important related work and it details my MSc thesis contributions, presenting the results of an extensive set of experiments made with the Yelp academic dataset, that attest for the adequacy of the proposed method. The experimental results used to attest the adequacy of the proposed method, showed that an increased usage of contextual information indeed results in an increased accuracy.

1 Introduction

Recommendation systems constitute nowadays an important sub-area of information retrieval. Typically, these systems are used on e-commerce services, and they try to predict the preference that a user would give to an item. The predictions are made with basis on the information submitted by the user, or by other users, and with these predictions the system can then recommend products or services that are potentially of interest to the user [12]. On the other hand, the recent emergence of online social networks provides us with large amounts of information about user behavior. Moreover, location-based social networks like Yelp register the local business that users frequently visit, providing huge datasets with information about the services that are preferred by the users. This work argues that this information can be used by a recommender system in order to discover new interesting business to be recommended to users.

In my MSc thesis I developed a recommendation method based on personalized random walks over a graph that encodes associations between users, items

and contextual attributes. The purpose of this method is to recommend locations to users based on data from location-based social networks such as Yelp, and the graph-based representation of the problem is constructed using the explicit feedback contained in the Yelp Academic dataset [1]. An adaptation of the Personalized PageRank algorithm was used for simulating the random walks, in order to recommend the top$-k$ local business to users. A series of experiments with the Yelp Academic dataset was made, testing different graph-based representations for the local business recommendation problem. These graph-based representations where as follows:

1. A graph encoding associations between users and locations (e.g., user visits).
2. A graph encoding user visits and associations from businesses to their locations (i.e., the locations of the businesses).
3. A graph encoding users visits, business locations, and associations from business to their respective categories.

Results from an extensive set of experiments have shown that an increased usage of contextual information indeed results in an increased accuracy.

The rest of this paper is organized as follows: Section 2 describes related work previously developed in the area. Section 3 explains the proposed recommendation method, describing also the graph representations that were considered. Section 4 presents the evaluation methodology and the results of an extensive set of experiments performed for testing the effectiveness of the proposed method, under different configurations. Finally, Section 5 summarizes the most important aspects of this paper, and presents directions for future work.

2 Related Work

Many recommendation systems have been developed. These systems recommend items which may be of interest to a considered user, with basis on user profiles. In the simplest formulation, recommendation methods predict the utility of an item, which is represented by a rating indicating how a user liked an item. These predictions are, for instance, based on the ratings given by the user to other items that he had interested with in the past [1].

Recommendation systems can be categorized into three different groups, namely content-based recommender systems, collaborative recommender systems and hybrid recommender systems. In content-based recommender systems, items are recommended to the users based on the content of the items that the user preferred in the past. These recommendations are made by matching user profiles with descriptions of the content of the items [4]. In the collaborative filtering approach, the opinions of the users are used to recommend new items that the user may like, and these recommendations are based on the similar preferences shared among a group of users [7]. The main idea of these systems

[1] `https://www.yelp.com/academic_dataset`

is that a group of users, who liked the same items in the past, probably share the same preferences. This group of users is called the *nearest neighbours*. In this way, recommendations can be made to a user of this group, suggesting him items that he never saw but that have been seen by his nearest neighbours, who share the same tastes and who liked items in common in the past [5].

Formally, let $p(c, s)$ be the preference of user c for an item s. This preference is estimated based on the preference $p(c_{nn}, s)$, assigned to item s by the nearest neighbours c_{nn} of the user c [1].

Collaborative recommender systems are currently the most used, and their main advantage is that items in different categories, even without similar content descriptions, can be recommended [5].

The final approach in the previous enumeration relates to hybrid recommender systems. These systems aim to combine multiple recommender techniques, or combine together multiple results of different recommender systems, to produce a single output. Therefore, hybrid systems exploit the advantages of different recommender systems, while avoiding their shortcomings [4].

The importance of profiles in content-based recommender systems is discussed by Cantador et al. [3]. The authors proposed a social tagging system where users create or upload items, annotate these items with tags, and share them with other users. A whole set of tags is commonly known as a *folksonomy*, and these sets are used to search and discover items of interest. Social tags are content features which describe both user and item profiles, which are defined in terms of weighted lists. Generally, users annotate, on their profiles, items which are relevant for them, so the tags they provide are assumed to describe user states, needs and preferences. Therefore, we can assume that the more a tag is employed by a user, the more important that tag is for the user. Moreover, the user's profiles indicate the number of times an item has been annotated with a tag, which usually describes their contents. The authors proposed three different schemes to weight the components of tag-based user and item profiles, and they also proposed for, each tag weighting scheme, five models to evaluate the similarities between user and item profiles [3].

One particular technique which has been widely used in recommendation systems is graph-ranking. This technique has been introduced into recommendation systems to model the interactions between users and items on a graph. Several graph-based recommendation methods have been introduced, such as the random walk method from Sangkeun et al. [8], which adapts the Personalized PageRank algorithm in order to rank entities. The authors modeled recommendation as an entity ranking problem, and they proposed to represent an implicit feedback dataset as a bipartite graph, latter using this graph for performing the recommendations. In this graph, a node is either an entity or a combination of entities, and the weights between nodes are assigned using the entity relationships implied from the dataset, namely the number of co-occurrences between entities. Then, the nodes are ranked according to a given query, which is also represented as a set of nodes. The ranking of nodes, and consequently the recommendation, is done by adapting the Personalized PageRank algorithm.

Recommender systems suggest to a user top ranked items with respect to a personalized ordering. The items are ranked according to a scoring algorithm, which ranks products or items for every given user, respecting the user expected preferences. In order to recommend the top ranked items to potentially interested users, Gori et al. proposed another random-walk based scoring algorithm called Item-Rank, which ranks products according to the expected user preferences [6]. The ItemRank algorithm is a biased version of the PageRank algorithm, which determines an importance score $PR(n)$ for every node n in a generic graph. Therefore, ItemRank scores induce a sorting of items according to their expected liking for a given user. The higher is the ItemRank for an item, the higher will be the probability that a user will prefer the item [6].

Capturing user preferences over time is a challenge in recommender systems. For instance Xiang et al. presented a graph-based recommendation method, which models user's long-term and short-term preferences over time [13]. This model, called Session-based Temporal Graph (STG), introduces session nodes which capture a user-specific time aspect. The graph is in this case a bipartite graph and it is created based on $< user, item >$ and $< session, item >$ associations. In STG, long-term interests are captured through user-item connections, and short-term interests through session-item connections [13]. Based on this framework, the authors proposed a new algorithm, named Injected Preference Fusion (IPF), to balance the impacts of user's time preferences for accurate recommendation. The idea of the IPF is to consider the user node n_u and its related session node $n_{u,t}$ as the source to be injected with user preferences. The preferences injected into the user node will be propagated to items $N(u)$ viewed by the user over long periods of time, and then they tend to propagate to unknown items approximate to $u's$ long-term preferences. The preferences injected in the session node will propagate to items $N(u,t)$ viewed by the user at time t, and then they tend to propagate to unknown items approximate to $u's$ short-term preferences. Consequently, the user preferences to an unknown item node is the sum of the weights of all incoming paths from both user and session nodes, and then the node with the highest preference is the one recommended to the user [13].

A particular application of recommendation systems relates to location-based social networks. These social services, such as Foursquare and Google Places, are available on mobile devices, and people can use them to trace, annotate, and share their experiences about the locations they visit, as they navigate through their daily lives. Users share their experiences, by notifying their friends of the place where they are through a *check-ins*. This means that users' mobility can be linked both to their social connections, as well as to the spatial layout of their cities. For instance Anastasios et al. proposed a recommendation approach named random walk around the city, which is a new venue recommendation system based in data from location-based social networks [10]. The authors considered a sample of check-in data over a pre-determined temporal period, from Foursquare and Gowalla. The data was represented in an undirected graph, where the nodes were users and venues, and where user i was linked to venue

j if the number of check-ins $c_{i,j}$ made by i on j was non-zero. Furthermore, a user was linked to another user if the pair were friends. Then, the authors used a personalized version of the PageRank algorithm, in order to calculate a rate in the random walk with restart to the node of the user. This rate was used in the recommendation of places to the users. The authors also introduced a weighted and directed version of the random walk approach. The proposed random walk with restarts favours places that are more connected to the user, through friends, through visited places, or through any combination of factors [10].

3 Recommendation with Random Walks

In many applications of recommender systems, the flexibility of the recommendation process is a very important property. This property is the capacity of handling multiple dimensions, as well as various recommendation types. However, the recommendation task has typically been considered as a problem of defining an utility function $f : User \times Item \rightarrow Utility$, where the estimation of the utility function f is done based on prior user preferences. This function is latter used to predict the utility of items to the given user [8]. Afterwards, the top$-k$ items with the highest utility are recommended to a given user [1]. However, it is not enough to assume that there are only the user and the item dimensions in a recommendation application. In most of the real world applications there are a range of other attributes which can be incorporated into the recommendation process. For example, in the case of my work, the task is to recommend interesting local business to users. Therefore, contextual attributes such as the location of the business, or even the category to which it belongs, can be important factors to take into account. Thus, considering only the typical form of recommendation is not enough, because other types of information can be very useful too, and they can improve the quality on the recommendation [8].

In order to incorporate flexibility and contextual information in the recommendation of local business to users, I propose a graph-based multidimensional recommendation method based on random-walks with personalized restarts. A node in the graph is either a user, a local business, or any contextual attribute of the local business. Positive weights between nodes corresponding to users and businesses are assigned using the relationships implied in the explicit feedback dataset. Then, the recommendation is considered as the problem of ranking business nodes according to a given personalized vector, that is also represented as a set of nodes. An adaptation of the personalized PageRank algorithm was used to rank the business nodes. This algorithm is a variation of the PageRank algorithm [11]. The PageRank algorithm is based on a process of random-walks with restarts on graphs. In its simplified version, the process is based on the stationary probability distribution of a random walk on the graph of the web. This process models the behaviour of a random surfer which keeps following successive links at random. However, if the surfer gets into a loop of nodes, it is unlikely that the surfer will continue in the loop forever. The surfer will randomly jump to some

other node. This behaviour can be modelled by having the surfer periodically getting bored and jumping to a random node for restarting [11]. This restart is done according to some transition probability during the random walk process. In the regular PageRank algorithm this transition probability is constant. Still, personalized and weighted versions of PageRank have been developed, in order to calculate the scores according to the user's interest. In the PageRank algorithm, the random walk process is defined by the following recursive definition:

$$PR\left(p_i\right) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR\left(p_j\right)}{L\left(p_j\right)} \tag{1}$$

In the formula, p_i is the node of the graph to be ranked, N is the total number of nodes, $M\left(p_i\right)$ is the set of nodes that are connected to p_i, $L\left(p_j\right)$ is the number of forward links of the node p_j, and $\frac{1-d}{N}$ is a residual probability where d is a damping factor usually set to be $d = 0.85$, that represents the uniform random jump with restart to the node p_i.

As said before, the personalized PageRank algorithm is an adaptation of the original PageRank algorithm. On the personalized PageRank, the random walks are personalized by a vector that normally represents a user's interest. In this case, the probability of jumping to a random node in the graph is determined according to this personalization vector. Thus, the personalized PageRank is calculated as follows:

$$PersonalizedPR\left(p_i\right) = (1-d)\,\frac{1}{N} s_{p_i} + d \sum_{p_j \in M(p_i)} \frac{PersonalizedPR\left(p_j\right)}{L\left(p_j\right)} \tag{2}$$

In the formula, s_{p_i} indicates the score of the node p_i in the personalization vector which defines the probability of restarting in a random walk [8].

My experiments involved the recommendation of interesting local business to users, based on the explicit feedback from the Yelp academic dataset. The data was represented as an undirected graph, where the nodes are users, business, or contextual information such as the business location and the business category. A user is connected to a business if the user has rated the business in the Yelp with a value $\geqslant 3$. To be more precise, in the Yelp users rate local business with a star value between one and five, where values bellow three indicate that the business is not of the interest of the user, so these values are considered negative values. Since, the personalized random walks only account with positive weights associated to the edges, the rates with a value bellow three were discarded. A relevance number, on a scale of $[1;3]$, was attributed to every rating given by a user to a business. Furthermore, for each user, a personalized vector was created, and this vector represents the user's interest for each business, to which the user is connected. The links between users and business were annotated with a weight, which influences the transition probability for the business that had higher ratings, instead of having an uniform probability on all the links that leave from a node. These weights were stored in each personalized vector. Latter, each

personalized vector was normalized by dividing the values, in each position in the array, by the sum of all the values contained in the array.

In order to see the effect of adding different types of contextual information into the graph representation, the personalized random walk recommendation method was tested over different types of graphs, namely:

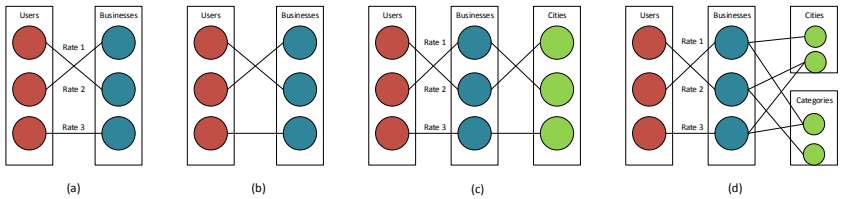

Fig. 1. Different graph representations for the Yelp academic dataset.

1. A bipartite graph with nodes corresponding to users and local business, with weighted undirected edges connecting users to the local business that they rated with a star score greater or equal to three;
2. A bipartite graph with nodes corresponding to users and local business, with a weight of one in the undirected edges connecting users to the local business;
3. A tripartite graph with nodes corresponding to users, local business and cities, with weighted edges connecting users to the local business that they rated with a star score greater or equal to three, and without weights in the undirected edges connecting local business to their corresponding cities;
4. A tripartite graph with nodes corresponding to users, local business, cities, and local business categories, with weighted edges connecting users to the local business that they rated with a star score greater or equal to three. No weights were used in the undirected edges connecting local business to their cities and to their corresponding categories.

Figure 1 provides an illustration for the four types of graph structures described on the previous page, where (a) illustrates Setting 1, (b) illustrates Setting 2, Setting 3 is illustrated with the graph shown in (c) , and Setting 4 with the graph in (d).

4 Experiments

Several experiments have been conducted in order to evaluate the proposed method, by comparing the obtained results across different settings. This section starts by describing the dataset that was used, as well as the evaluation measures used to acess the quality of the results. Finally, the section explains the experiments that were carried out, and the results that were obtained.

4.1 Dataset and Evaluation Measures

The experiments conducted in this work used Yelp's Academic Dataset. This dataset consists of reviews given by user's to local business. It contains 152326 reviews of 65888 users, on 6899 business. Using the dataset, a graph was constructed and one thousand users were selected for testing, based on the ones who had the highest degree, this is the users who had a larger number of edges linking them to business. The reviews of these one thousand users were considered for generating a test set.

Most recommender systems use the opinions of a group of users to help other users identifying contents of interest to them. Evaluating these systems and their algorithms can be a very difficult task, since evaluation has to take into account which attributes should be measured and which metrics should be used in each attribute. Therefore, and since the focus was only on the quality of the top-10 ranked business, the metrics used were the Precision@10, Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG)@10.

Precision is defined as the ratio between the relevant items selected and the number of items selected [2], as shown bellow:

$$P = \frac{N_{rs}}{N_s}. \tag{3}$$

Precision takes all the returned items into account, but it can also be evaluated at a given cut-off rank, considering only the top-k results returned. This is called Precision at k or $P@k$. Only the first 10 items within the top-k results returned by the system were considered in our case.

The MAP computes the average precision across several different levels of recall. Formally, if the set of relevant items for an information need $q_j \in Q$ is $\{i_1, \cdots, i_{mj}\}$, and R_{jk} is the set of ranked retrieval results from the top result, until item i_k is returned, then the MAP is given by the formula bellow [9]:

$$\text{MAP}\,(I) = \frac{1}{|I|} \sum_{j=1}^{|I|} \frac{1}{mj} \sum_{k=1}^{mj} P(R_{jk}). \tag{4}$$

When a relevant item is not retrieved, the precision value is taken to be zero. Notice that MAP takes into account the entire list of retrieved results.

The NDCG measures the gain or the usefulness of a item, based on the position of the item in a result list. Similarly to the $P@k$, NDCG evaluates over some number k of the top results, and it is given by the following formula:

$$\text{NDCG}\,(I, k) = \frac{1}{|I|} \sum_{j=1}^{|I|} Z_{kj} \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2 (1 + i)}. \tag{5}$$

In the formula, I is a set of items, rel_i is the relevance score evaluators gave to a item i, and Z_{kj} is a normalization factor calculated with basis as a perfect ranking's NDCG @K [9]. Thus, similarly to what was done in the P@10 case, only the first 10 items with the top-k results returned were considered.

4.2 Experimental Results

	Precision@10	MAP	NDCG@10
Users and Businesses	0.8505	0.9923	0.7726
Weighted Graph with Users and Businesses	0.8506	0.9927	0.7515
Users, Businesses and Categories	0.8536	0.9963	0.7519
Users, Businesses, Categories and Cities	0.8508	0.9931	0.7515

Table 1. Results obtained with different graph-based representations.

After characterizing the dataset, we began to compare the proposed recommendation method over the four graph-based representations, with a parameter of 40 iterations for computing the steady-state probabilities of the random walk, and the value $d = 0.8$. An evaluation methodology was used in which, using the dataset, a graph was constructed and 1000 users were selected based on the ones who had the highest degree, this is the users who had a larger number of edges linking them to business. Recommendations were generated for this set of 1000 users with more ratings in the Yelp Academic, using half of their ratings to build the personalization vector and the other half for measuring the results. In order to quantify the quality of the generated recommendations, three evaluations metrics were used. These evaluations metrics are Precision@10, the Mean Average Precision (MAP), and the Normalized Discount Cumulative Gain, also at a cutoff 10 ($MDCG$@10).

Table 1 presents the results obtained when comparing the four different graph representation. The results show that the inclusion of the contextual information results in an increase in the recommendation accuracy. The best results correspond to the representation which considers the nodes corresponding to users, business, business categories.

Besides analyzing the performance of the recommendation over the entire set of 1000 users in the four different graph-based representations, I also analysed the effect of varying the number of iterations on the PageRank algorithm. Also, the effect of varying the value of the probability of a random restart was analyzed. Figure 2 shows the effects of changing these parameters, when considering the graph with nodes corresponding to users and local business, with weighted undirected edges. Figure 2 shows that when changing the number of iterations, the results stabilize within a small number of iterations. Figure 2 also shows that accuracy decreases if the restart probability is greater than 0.8, showing

that taking longer random-walks in the graph can start to deviate the rankings towards incorrect results.

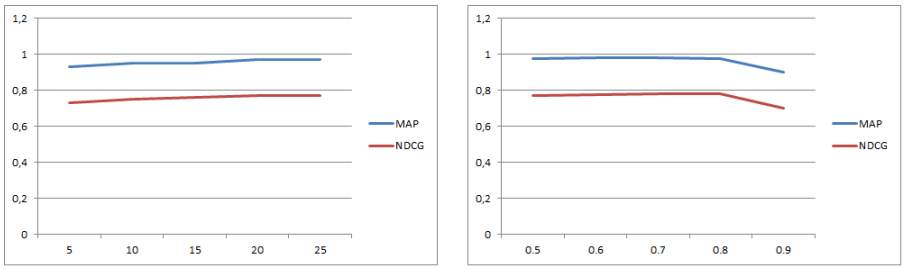

Fig. 2. Effect on results when changing the number of iterations or restart probability.

5 Conclusions and Future Work

The generality of the research in recommendation systems has been focused in improving accuracy, diversity and novelty. However, the effort to include flexibility in recommender systems has been very little. In most of the real world applications there are a range of attributes which can be incorporated into the recommendation process. Therefore, the flexibility of the recommendation process is a very important property in many applications of recommender systems, because it deals with multidimensional information. This is important not only for the accuracy of recommendation, but also for many other aspects, such as the performance of recommendation.

In my MSc thesis, I developed a recommendation method based on personalized random walks, which involved the recommendation of interesting local business to users of the Yelp location-based social network. This recommendation was done through the usage of a graph-based representation of the available information. The use of a graph data model in the recommendation process has several advantages. Any type of information can be modeled as nodes in a graph, and various types of contextual information can easily be incorporated into the recommendation method.

The results of several experiments with the Yelp academic dataset showed that the introduction of contextual information results in an increase in the accuracy. The best results correspond to a representation which considers nodes corresponding to users, businesses and business categories, with weighted edges between users and businesses.

Exploiting contextual information can improve the accuracy of recommender systems. In this sense, many promising directions are now available for future work. Even in the same context of recommending interesting local business to users, it would be interesting to see if other types of contextual information

would also help to increase the recommendation accuracy, such as friendship and the proximity associations between the users.

References

1. G. Andomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 17(6), 2005.
2. F. Cacheda, V. Carneiro, D. Fernández, and V. Formoso. Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Transactions on the Web*, 5(1), 2011.
3. I. Cantador, A. Bellogín, and D. Vallet. Content-based recommendation in social tagging systems. In *Proceedings of the 4th ACM Conference on Recommender Systems*. ACM, 2010.
4. Şule Gündüz Ögüdücü. Web page recommendation models: Theory and algorithms. *Synthesis Lectures on Data Management*, 2(1), 2010.
5. M. de Gemmis, L. Iaquinta, P. Lops, C. Musto, F. Narducci, and G. Semeraro. Learning preference models in recommender systems. In *Preference Learning*. Springer-Verlag, 2010.
6. M. Gori and A. Pucci. A random-walk based scoring algorithm with application to recommender systems for large-scale e-commerce. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2006.
7. J. Herlocker, J. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*. ACM, 2000.
8. S. Lee, S. il Song, M. Kahng, D. Lee, and S. goo Leel. Random walk based entity ranking on graph for multidimensional recommendation. In *Proceedings of the 5th ACM Conference on Recommender Systems*. ACM, 2011.
9. C. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
10. A. Noulas, S. Scellato, N. Lathia, and C. Mascolo. A random walk around the city: New venue recommendation in location-based social networks. In *Proceedings of the IEEE International Conference on Social Computing*, 2012.
11. L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the Web. In *Proceedings of the 7th International World Wide Web Conference*, 1998.
12. F. Ricci, L. Rokach, B. Shapira, and P. Kantor. *Recommender Systems Handbook*. Springer US, 2011.
13. L. Xiang, Q. Yuan, S. Zhao, L. Chen, X. Zhang, Q. Yang, and J. Sun. Temporal recommendation on graphs via long- and short-term preference fusion. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2010.