An Entity Graph Based Recommender System

Sneha Chaudhari, Amos Azaria and Tom Mitchell
School of Computer Science, Carnegie Mellon University
Pittsburgh, PA

ABSTRACT

We propose a recommender system which exploits relations present between entities appearing in content from user's history and entities appearing in candidate content. In order to identify such relations, we use the knowledge graph of NELL, which encodes entities and their relations. We present a novel normalized version of Personalized PageRank, to rank candidate content. We test our approach on the movie recommendation domain and show that the proposed method outperforms other baseline methods, including the standard Personalized PageRank.

1. INTRODUCTION

In this paper we present the Relation of Entities Recommendation Agent (RERA), a new content-based system which takes into account relations which may exist between entities which appear in the users past consumed content and entities which appear in the suggested content. For example, consider the following scenario. Suppose some user has read articles about a football team "Real Madrid" in the past. The recommender system needs to decide whether to recommend a new news article which has mentions of "James Rodriguez" in it. A relation between "James Rodriguez" and "Real Madrid" ("James Rodriguez" plays for the team "Real Madrid"), can hint that this new news article is highly relevant. RERA belongs to a family of recommender systems which take use of a knowledge base such as DBpedia (e.g. [6, 7, 5]).

RERA uses a knowledge graph in which the nodes are real world entities (e.g. "James Rodriguez") and the edges are the relations between them (e.g. PlaysOnTeam(James Rodriguez, Real Madrid)). RERA extracts the entities which appear in the content consumed by the user (or those which appear in the description of the items purchased by the user); this first set of entities is assumed to be the user's interests. Once RERA is required to recommend new items, it extracts all entities which appear in each of the proposed new items. RERA then ranks these new items according to relations which exist in the knowledge graph between the entities in each of these items and the entities in the user's interests. RERA does not require any information on other users' preferences (as required by other graph based approaches such as [5]). We have enriched the Never Ending Language Lerner (NELL) [4] with Subject-Verb-Object (SVO) triples from the ClueWeb 2009 dataset [2] to serve as RERA's knowledge graph. In order to rank the suggested content, RERA uses a novel raking method based which takes into account both the PageRank of each entity and its *Personalized* PageRank (PPR) [3]. We test RERA in a movie domain (using the MovieLens data-set, combined with movie synopsises crawled from IMDB) and show that RERA, using our novel version of PPR, outperforms PPR and other baseline methods.

RecSys '16, September 15 - 19, 2016, Boston , MA, USA.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

2. RELATION OF ENTITIES RECOMMENDATION AGENT (RERA)

RERA receives as input a set of documents (which could be movie synopsis, new articles or item descriptions) which the user either liked or was interested in, in the past. RERA assumes that each item is basically a set (or bag) of NELL entities mentioned in the article. At first, RERA extracts noun phrases from these item descriptions using Stanford CoreNLP. Then each of these noun phrases are given as input to NELL's knowledge on demand API to extract the corresponding NELL entities. This set is assumed to be the entities which the user is interested in, or the set of the user's interests. In order to provide recommendations, RERA extracts NELL entities from the proposed content using the same method. The basic intuition behind RERA is that if the NELL entities present in the document are well connected with the NELL entities of the user interest, then the content should be highly relevant.

We consider three different methods, each method provides a score to each of the proposed documents. The documents are then ranked according to the score in a descending order (the higher the better). The first method is PageRank (PR). This method, in fact does not consider the user interest set at all. The simple PageRank algorithm computes a score for each node in the graph which denotes a rough estimate of how important the node is, or equivalently, what is the probability of visiting that node in a random walk. This method recommends the documents with the highest average PageRank score (the average is computed over all entities in every document).

The second method, is a variant of Personalized Page Rank (PPR). The motivation behind PPR, is that while the web surfer can still jump to a random page, it is (more) likely to jump to a page according to his or her unique preferences according to a normalized vector which encodes these preferences. In this method, the recommender system recommends the items with the highest average PPR score.

The third method takes into account both PR and PPR. Both PR and PPR have drawbacks. The problem with the PR method is that it does not take the user's preferences into account at all, and as-

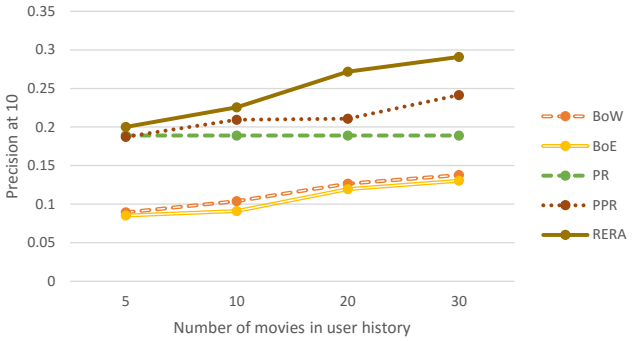

Figure 1: Precision at 10 for each of the methods, when predicting whether the user would like a movie, with movies in user history (training set) between 5 to 30

sumes all users are the same. While PPR is personalized, it ignores the fact that entities which get a high PPR score, may obtain such a score not necessarily because they are highly related to the user interests, but because that they appear to be "important" in the graph in general, i.e., have a high PR score in the first place. For example, assume that the entity "Sunday" appears in a document. This entity is likely to receive a high PPR score since many entities are connected to "Sunday" and therefore, it is very likely that many of the user's interests are either directly connected to "Sunday" or connected via other entities. However, since it is so highly connected, it is also likely to receive a high PR score. The same argument holds for an entity which receives a low score, but may be quite relevant to the user. For example, if an entity is connected only to one other entity, but that other entity happens to be in the user's interest set. Therefore, to overcome these difficulties, RERA uses PR to *normalize* the PPR score and the score for RERA is given by PPR/PR. RERA recommends the items with the highest average PPR/PR score.

3. EXPERIMENTAL EVALUATION

We tested the efficiency on RERA on the MovieLens 1M Dataset. The movie synopsis were crawled from IMDB. We split the movies rated by the users into train and test sets and measure the precision each of the methods reaches when recommending 10 movies. We use this metric since the top few recommendations are the most important for recommendation systems.

We considered the three methods mentioned in section 2, i.e., PR, PPR and RERA, along with two additional baselines: Bag-of-Entities (BoE), which considers only exact matches between the entities in the document and the user's interest, and the commonly used baseline of Bag-of-Words (BoW), which uses all words in the document (both for user interest and for proposed content), as opposed to using only the user entities. For BoW, we processed the documents for tokenization, stop-word removal and stemming. In the MovieLens dataset, we considered any movie which received 4 or 5 stars as a movie which the user liked and filtered users such that they liked or rated at least 50 movies. The input to the different methods (the training set which was used to compute the users' interests) varied from only 5 movies, to 30 movies. Precision at 10 values were averaged over 10 random trials.

Figure 1 presents the precision of each of the methods when recommending 10 movies, i.e., the fraction of movies in the top 10 scores which appear in the test set as those which were liked by the user. As depicted by the figure, RERA (PPR/PR) outperformed all baselines in all conditions, with a great increase in precision. PPR comes in second, and PR third, with the two baselines (BoE and BoW) way behind. As expected, the performance of all methods generally increases as the number of movies in the training set increases. The only exception is the PR method which does not take into account the user history at all.

Another interesting result which can be observed from Figure 1 is that BoW had only a very slight advantage over BoE. This is very encouraging, as this might hint that the entities in the document do indeed capture the essence of it, and therefore, recommending content based on the entities alone is as good as using all the words in the document.

4. CONCLUSION AND FUTURE WORK

In this paper we introduce RERA, a recommender system which uses an enhanced NELL knowledge graph consisting of entities and relations between them to recommend content to users. RERA finds the NELL entities that are of interest to the user and the NELL entities which are mentioned in the proposed content. RERA uses a novel enhanced version of the personalized page rank algorithm, to determine how well connected these sets of entities are in order to rank the relevance of the proposed content. We show that RERA outperforms other baseline methods.

Although our experiments were conducted in the movie domain, our approach is general and can work in any domain. News recommendation may be a better fit for our approach and we are currently working on methods to obtain data which we could experiment with. Our method could be used also for search. If a search-engine is aware of the user's general interest, RERA can help by providing an additional input for the ranking algorithm.

5. ACKNOWLEDGMENT

This research was supported by Yahoo! as part of the InMind project [1].

6. REFERENCES

[1] Amos Azaria and Jason Hong. Recommender system with personality. In *Proceedings of the 10th ACM conference on Recommender systems*. ACM, 2016.

[2] Jamie Callan, Mark Hoy, Changkuk Yoo, and Le Zhao. Clueweb09 data set, 2009.

[3] Soumen Chakrabarti. Dynamic personalized pagerank in entity-relation graphs. In *Proceedings of the 16th international conference on World Wide Web*, pages 571–580. ACM, 2007.

[4] Tom M Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Justin Betteridge, Andrew Carlson, Bhavana Dalvi Mishra, Matthew Gardner, Bryan Kisiel, Jayant Krishnamurthy, et al. Never-ending learning. In *AAAI'15*, 2015.

[5] Cataldo Musto, Pierpaolo Basile, Marco de Gemmis, Pasquale Lops, Giovanni Semeraro, and Simone Rutigliano. Automatic selection of linked open data features in graph-based recommender systems. *CBRecSys*, 15:10–13.

[6] Vito Claudio Ostuni, Tommaso Di Noia, Eugenio Di Sciascio, and Roberto Mirizzi. Top-n recommendations from implicit feedback leveraging linked open data. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 85–92. ACM, 2013.

[7] Alexandre Passant. dbrecâĂŤmusic recommendations using dbpedia. In *International Semantic Web Conference*, pages 209–224. Springer, 2010.