SemStim at the LOD-RecSys 2014 challenge

Benjamin Heitmann and Conor Hayes

INSIGHT @ NUI Galway
National University of Ireland, Galway
Galway, Ireland
firstname.lastname@insight-centre.org

Abstract. SemStim is a graph-based recommendation algorithm which is based on Spreading Activation and adds targeted activation and duration constraints. SemStim is not affected by data sparsity, the cold-start problem or data quality issues beyond the linking of items to DBpedia. The overall results show that the performance of SemStim for the diversity task of the challenge is comparable to the other participants, as it took 3rd place out of 12 participants with 0.0413 F1@20 and 0.476 ILD@20. In addition, as SemStim has been designed for the requirements of cross-domain recommendations with different target and source domains, this shows that SemStim can also provide competitive single-domain recommendations.

1 Introduction

In this paper we describe our contribution to the "Linked Open Data-enabled Recommender Systems" challenge at the Extended Semantic Web Conference (ESWC) 2014. The recommender system which we built in order to participate at the challenge employs our SemStim algorithm.

SemStim is a graph-based recommendation algorithm which uses Linked Data from DBpedia in order to provide recommendations. Our approach is innovative due to the three main **design choices** of our algorithm: (1) It is not affected by the *sparsity of rating data* or by the *cold-start problem* (cf. Schein et al. [1]), as it only requires user preferences of the user for whom recommendations are being generated. In other words, it will even work when there is only a single user in the system. (2) It is not dependent on *availability or quality of meta-data* about recommendable items, beyond the quality of linking items to DBpedia. It only makes use of DBpedia as a graph. Other features of Linked Data such as reasoning or the content of literals are not used. (3) Our algorithm has been designed to generate recommendations in a target domain based on user interests in a different source domain, by using indirect connections from DBpedia between items of the two domains. This enables us to provide *cross-domain recommendations* as defined by Fernandez-Tobias et al. [2], even in the worst-case scenario, when there is no overlap between users and items of the two domains.

All three evaluation tasks of the challenge are single-domain recommendation tasks, as both the user preferences and the recommendable items are taken from the provided DBbook dataset. Therefore, the *goal* of our participation in the challenge, is to show that SemStim has a competitive performance for single-domain recommendation, despite the disadvantage of being designed for the requirements of cross-domain recommendation.

2 High-level overview of the SemStim algorithm

Our SemStim algorithm is an enhanced version of spreading activation (SA) as described by Crestani in [3]. SA enables finding related entities in a semantic network in a fuzzy way without using reasoning, while still being deterministic and taking the semantics of the network into account. SemStim extends basic SA, by adding (1) *targeted activation*, which allows us to describe the target domain of the personalisation, and (2) *constraints* to control the algorithm *duration*. Due to the space constraints of this paper, we will only provide a high-level overview of the algorithm. However, the publication of a detailed, formal description is forthcoming.

SemStim spreads activation on a graph using the RDF data model, so we define the graph $G = (V, E)$, with V as the set of nodes $V = \{v_1, \ldots, v_n\}$ used as vertices in the graph, and $E = \{e_1, \ldots, e_m\}$ as the set of predicates used as edges in the graph.

We denote each iteration of the algorithm as *wave* $w \in \mathbb{N}_0$, and each restart of the algorithm as a *phase* $p \in \mathbb{N}_0$. The *activation state* of the graph G in wave w is denoted by $\mathbf{a}^{(w)} \in \mathbb{R}^n$, with $\mathbf{a}_v^{(w)}$ as the activation of node $v \in V$. The *source domain* $S = \{s_1, \ldots, s_f | s_i \in V\}$ is the subset of V from which items in the user profiles are taken, e.g. all URIs from DBbooks on DBpedia. The *target domain* $D = \{d_1, \ldots, d_e | d_i \in V\}$ is the set of nodes which represent the recommendable items among all of the nodes in V.

Each *user profile* P is a set of nodes $P = \{p_1, \ldots, p_k | p_i \in S\}$. τ is the activation threshold of a node, which determines how much activation a node needs to accumulate before it can fire. Then the *initial activation state* $\mathbf{a}^{(0)}$ is defined by setting the initial activation of all nodes in the user profile P to τ, so that nodes in the user profile can immediately fire and contribute to the activation state $\mathbf{a}^{(1)}$ in wave 1.

For each iteration w of the algorithm, and all nodes $v \in V$, the activation state $\mathbf{a}_v^{(w)}$ with $w > 0$, is obtained from the previous state $\mathbf{a}_v^{(w-1)}$ by first applying the input function $I(v, w)$ to v. Then the activation function $A(v, w)$ is applied to the result of $I(v, w)$. As the last step, the output function $O(v, w)$ is applied to the result of $A(v, w)$.

The *input function* $I(v, w)$ aggregates the weighted output of the direct neighbors of node v in wave w. The direction of an edge in the graph is not taken into account, resulting in an undirected view on the graph. The *activation function* $A(v, w)$ determines the amount of activation which a node can spread to its neighbors when it reaches its activation threshold. The *output function* $O(v, w)$ determines if a node can fire in wave w. This is dependent not just on the activation function $A(v, w)$, but also on binary functions which determine if the node can fire, or if the algorithm has reached the termination condition.

The $Restart(\mathbf{a}^{(w)})$ function determines if a restart is necessary. Whenever the number of activated nodes stays the same between wave $w - 1$ and wave w, then no further activation is possible without restarting by firing all activated nodes again. Each restart marks the start of a new *phase* of the algorithm.

The $Fire(v, w)$ function determines if a specific node v in wave w can spread the result of its activation function to its neighbours. There are 2 conditions for firing: (1) The accumulated activation of the node was below the activation threshold τ in the previous wave $w - 1$, and has reached τ in the current wave w. (2) If

the $Restart(\mathbf{a}^{(w)})$ function signals a restart, and if the total number of restarts is below the maximum number of phases ρ_{max}. Note that each node can only cross the activation threshold once, while it can be restarted up to ρ_{max} times.

The $Terminate(\mathbf{a}^{(w)})$ function determines if the algorithm as a whole can terminate. There are two termination conditions: (1) the number of activated nodes from the target domain D is higher then the required number of targets, which is given as θ. Or (2) if the algorithm has reached the maximum number of waves, which is specified by w_{max}.

The *top-k recommendations* for target domain D are determined after the termination by sorting the set $\{\mathbf{a}_v^{(w)} > \tau \mid v \in D\}$ of all activated nodes from the target domain by their activation values, and returning the first k items.

Please note that SemStim uses a non-linear activation model. This differentiates it from other variations of SA, such as the linear activation model, which is described in Berthold et al. [4]. The linear activation model introduces the problem of making SA "query independent", as the SA algorithm then converges on the principal eigenvector of the weighted adjacency matrix [4] independent of the user preferences. In order to avoid this problem and allow making personalised recommendations for each individual user, SemStim uses input, activation and output functions which constrain the activation and which implement a non-linear activation model.

3 System description

3.1 Selection of evaluation tasks

The challenge provides data for 3 evaluation tasks, which are (1) the rating prediction task, (2) the top-N recommendation task and (3) the diversity recommendation task. As we now explain, SemStim is only applicable to the diversity recommendation task.

SemStim is not applicable to the *rating prediction task*, as our algorithm does not make use of rating data. In addition, SemStim could not perform the *top-N recommendation task*, given the properties of the evaluation data set. We analysed the test data set, and we found that the median number of items in the test profiles is 12. In order to perform the given top-N recommendation task, the items in the test profile need to be ranked and the top-5 items need to be recommended. SemStim would need to start with the train profile of a user and then find at least 5 books from the 12 books in the test profile in a graph with 11 million vertices. Initial testing showed that this exceeds reasonable limits for runtime and memory, so we were not able to participate in the top-N recommendation task.

However, we were able to apply SemStim to the given *diversity recommendation task*. In order to perform the diversity recommendation task, a list of top-20 recommendations needs to be made, using the unrated books of a user. As the DBbook data set contains 6733 books, this amounts to using SemStim to find 20 of 6733 books on the DBpedia graph for 6181 users. Testing showed that using the available data, SemStim can perform the diversity recommendation task for all 6181 users.

3.2 Applying SemStim to the diversity recommendation task

We use SemStim to generate 20 recommendations for each user profile in the training data of the diversity recommendation task in 6 steps, as follows:

1. We determine the set P of start nodes for the active user. The train data contains binary ratings, where 1 indicates relevance for the user and 0 indicates irrelevance. If the user has any positive preferences, then their corresponding DBpedia URIs are added to P, while ignoring all negative preferences. However, if the user has no positive preferences at all, then the DBpedia URIs of all his negative preferences are added to P. 1463 users have only negative preferences in their user profiles.
2. We set the target domain T to be the set of DBpedia URIs for all unrated books of the active user.
3. We set the constants of SemStim. We experimentally determined that we achieve the best results for the following configuration: activation threshold: $\tau = 0.7$; maximum number of waves: $w_{max} = 1$; maximum number of phases: $\rho_{max} = 5$; required number of activated nodes from T: $\theta = 20$; default weight of predicates: $\beta = 1.0$; initial node output before applying modifiers: $\alpha = 4.0$.
4. Then we run the algorithm. When the algorithm terminates, we rank the activated nodes by their activation value in descending order, and return the corresponding DBbook IDs of the first 20 nodes.
5. If the number of activated nodes is less than 20, we add random, unrated items to reach the target size of 20, as the evaluation system requires all users to have 20 recommendations.

3.3 Usage of Linked Data

We use a subset of DBpedia 3.8 with 67 million edges and 11 million vertices, which includes all article categories, disambiguation links, instance types, mapping-based properties, transitive redirects, categories, and entity types. We choose this subset, because SemStim as a graph algorithm requires as much available data about edges between vertices on DBpedia as possible, and this subset contains all the available edge data. Conversely we did not use any literals from DBpedia, as they are not used by SemStim. The linkage data which connects the books in the DBbook data of the challenge to DBpedia, has been provided by the challenge organisers.

In order to store this subset of DBpedia, we used the Header-Dictionary-Triple (HDT) store [5], which provides a compact data structure and binary serialisation format for RDF. HDT keeps big datasets compressed in memory while allowing read-only access to the graph without prior decompression. The HDT serialisation of our DBpedia subset takes up 559 MB. The average duration of a simple subject or object query using HDT on this data set is 0.2 milliseconds.

3.4 Hardware infrastructure

To run the diversity evaluation task, we used a server with 24 Intel Xeon cores at 2.40GHz and with 96 GB RAM. We ran 12 threads concurrently, the remaining capacity was used for e.g. garbage collection, and by other users of the server. The duration of executing the diversity evaluation task is around 4 hours.

4 Lessons learned

The overall results show that SemStim has a competitive performance which is comparable to that of the other participants in the diversity task. This is supported by SemStim taking the 3rd place out of 12 participants in the leader board of the diversity recommendation task. This is based on the average of the rankings on F1-score and inter-list diversity (ILD). The F1-score of SemStim was ranked in 4th place with 0.0413, and the ILD of SemStim was ranked in 7th place with 0.476.

Further, we found out that our algorithm is not over-optimised for either F1-score or diversity, as the parameters of the algorithm can be tuned towards either of these metrics. Independent of the algorithm parameters, the baseline of the ILD diversity is 0.4570. By setting the activation threshold very low at 0.1, and setting a maximum of 4 phases and 3 waves, SemStim can achieve an F1-score of 0.0593, which comes at the cost of a comparatively low ILD of 0.4591. On the other hand, SemStim can achieve a high ILD of 0.4858, by setting a higher threshold of 0.6 and using only a maximum of 1 phase and 1 wave. However this is at the expense of a low F1 score of 0.0019.

5 Conclusions

In this paper we described our contribution to the "Linked Open Data-enabled Recommender Systems" challenge at the Extended Semantic Web Conference (ESWC) 2014. By participating in the challenge we showed that SemStim has a competitive performance which is comparable to that of the other participants in the diversity task, as SemStim took the 3rd place out of 12 participants. This shows that SemStim can provide competitive recommendations for the single domain recommendation task, although the algorithm was designed for the requirements of cross-domain recommendations.

Acknowledgements: This publication has emanated from research supported in part by a research grant from Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289.

References

1. Schein, A.I., Popescul, A., H., L., Popescul, R., Ungar, L.H., Pennock, D.M.: Methods and metrics for cold-start recommendations. In: Conference on Research and Development in Information Retrieval, ACM Press (2002) 253–260
2. Fernández-Tobías, I., Cantador, I., Kaminskas, M., Ricci, F.: Cross-domain recommender systems: A survey of the state of the art. In: Spanish Conference on Information Retrieval. (2012)
3. Crestani, F.: Application of spreading activation techniques in information retrieval. Artificial Intelligence Review **11**(6) (1997) 453–482
4. Berthold, M., Brandes, U., Kötter, T., Mader, M., Nagel, U., Thiel, K.: Pure spreading activation is pointless. In: Conf. on Information and knowledge management. (2009)
5. Fernández, J.D., Martínez-Prieto, M.A., Gutiérrez, C., Polleres, A., Arias, M.: Binary RDF Representation for Publication and Exchange (HDT). Web Semantics **19**(2) (2013)