Recommendations on a Knowledge Graph

László Grad-Gyenge*
laszlo.grad-gyenge@tuwien.ac.at

Peter Filzmoser†
peter.filzmoser@tuwien.ac.at

Hannes Werthner‡
hannes.werthner@tuwien.ac.at

Abstract

Most recommender system methods derive the user preferences from predefined information sources. For example, collaborative filtering is based on user rating values on items. Predefining information sources constrains the quality of the recommendation result by restricting the amount of information the recommendation method can operate on. In this paper we introduce an adaptive rating estimation method, which is capable to incorporate heterogeneous information sources and improves the recommendation quality.

To represent heterogeneous information, a graph based knowledge base is introduced. Recommendations are calculated with our novel method, recommendation spreading. Comparing recommendation spreading to collaborative filtering on the MovieLens 1M dataset shows that our method is able to combine heterogeneous information sources to provide higher coverage and the same rating estimation error. Furthermore, recommendation spreading is a potential method to overcome the cold start problem.

1 Introduction.

To enhance recommendation quality, several information sources have been involved into the recommendation process by the recommender systems community. Most of the methods we have found during our research explicitly define the type of information sources to calculate the recommendations from. To mention the most popular ones, recommendations can be derived from user preferences on items, user attributes, product attributes, social network, product description, user interaction, ontology information, purchase history or expert knowledge. One of our goals is to develop a recommendation framework providing an information representation method which is general enough to represent and to integrate information from various types of information sources. Our intention is to provide an information representation method which can act as a stable basis for the elaboration of more general recommender systems. By generality we mean methods, which are capable to weigh appropriately the available information source types instead of working with predefined ones.

The Machine Learning community has evolved several adaptive classification, clustering and prediction techniques. By adaptiveness we mean that after a certain period of training the method learns the underlying structure or the features of the data and makes its predictions and decisions based on the previously learnt model. By defining the knowledge base to contain a variety of information sources, our intention is to introduce recommendation methods which are influenced by machine learning in the sense they are adaptive and can be trained on past data. We think that this area is a promising direction of research. In this paper we present our first, spreading activation based calculation method, which is a promising step in this direction. Spreading activation is a rarely used technique in this field. For example Hussein et al. utilize activation spreading to deliver explanations to the user [10].

In this paper we will introduce our heterogeneous knowledge base which is capable to incorporate several information source types. The knowledge base is introduced as a directed, labeled, restricted hypergraph. We also define a recommendation calculation method which provides higher quality recommendations than collaborative filtering does. This way we prove on a working example that the involvement of an increased and heterogeneous amount of information sources can lead to higher quality recommendations.

Section 2 contains an overview of recommendation techniques, which operate with graph related representation methods. In Section 3 we provide a formal definition of our knowledge base. Section 4 describes our calculation method, which is based on spreading activation. In Section 5 we describe how we evaluated our pivot method and present our evaluation results. Section 6 concludes the paper and gives an insight into our plans for the future.

*Electronic Commerce Group, Vienna University of Technology

†Department of Statistics and Probability Theory, Vienna University of Technology

‡Electronic Commerce Group, Vienna University of Technology

2 Graph Based Representation.

A trend is visible to graph based representation in recommender systems. Collaborative filtering is the best-known recommendation method. In most cases the knowledge base of collaborative filtering is modeled with a matrix [16]. Collaborative filtering estimates user preferences on items based on the already known preference values with rather good results. Such preferences can be explicit (user rating, user like, purchase) or implicit (commenting an item, viewing item details, mentioning an item in a post). A visible trend in the field of recommender systems is to involve additional information sources to for example collaborative filtering in order to improve its performance. To represent this information, various methods were developed.

Konstas et al. [13] recommend music tracks for users. To improve the recommendation quality, in addition to item rating they involve user issued tags and social relationships between users into the knowledge base. They represent the different kind of relations as `UU` (user-user), `UTr` (user-track), `UTg` (user-tag) and `TrTg` (track-tag) with a partitioned matrix. Regarding information type content, each of those partitions of the matrix are homogeneous but the overall matrix is heterogeneous. Hidasi et al. [9] take context information into the recommendation process. To represent the information they utilize a tensor algebra, which can be treated as the generalization of matrices into higher dimension. We would like to mention here that in both of the above mentioned cases the matrix representation can be substituted with an equivalent, graph based representation.

Kazienko et al. [12] use a multi-layered graph to represent the information necessary to compute recommendations. To estimate user preferences, their method relies on contact lists, tags, groups, favorites, opinions and social networks. They represent each information type on a separate layer, where each layer contains a graph representing homogeneous information. The layered approach could be mapped to a unified but heterogeneous graph by adding a type attribute to the respective graph nodes and edges.

Furthermore, involving trust networks into the recommendation process was a visible trend of the recommender community in the years 2004-2006 [5][17][14][11]. A straightforward representation of a trust network is a directed graph. A directed graph would encode the users as nodes and the trust relationships as edges. Because of its explicitness, the involvement of this kind of information source has a good chance to increase recommendation quality. Trust networks can be seen as a subconcept of the more general, social network based relation, which however also increases the recommendation quality [6][13][7]. Social relationships also influence the shopping behaviour of people, which mechanism relates to the strengths of weak ties in networks [3]. The difference between trust networks and symmetric social networks is somehow similar to conditional and joint probability. In the case of trust network the information (trust) is represented in a directed graph, while in the case of a social network, social relationships are represented with non-directed edges. However, asymmetric social relationships are also a useful information source [4].

3 The Graph.

An important aspect of our research is to keep the information representation method as general as possible. This way we have the opportunity to work with various calculation methods in the future. We focus on adaptive methods where we can adapt the weights belonging to different information types according to new information available. On the other side, our intention is to present as much information as possible in order not to constrain the coverage and precision of the recommendation methods.

3.1 Definition. We introduce our representation method as a labeled, weighted, restricted hypergraph, as

$$\mathcal{K} = (N, E, T_N, T_E, t_N, t_E, w_E, A, a_n, a_e)$$

. N represents the set of nodes existing in the graph, $E \subseteq \{\{u, v\} | u \in N \wedge v \in N\}$ represents the set of edges between the nodes. T_N is the set of node types, T_E is the set of edge types. Function $t_N \subset N \times T_N$ assigns a node type to each node, function $t_E \subset E \times T_E$ assigns an edge type to each edge. Function $w_E \subset E \times \mathbf{R}$ assigns weights to the edges. A is a set containing attribute values, which can be assigned to nodes or edges. Function $a_n \subset N \times A$ assigns attribute values to nodes, function $a_e \subset E \times A$ assigns attribute values to edges.

Node types define the type of entities represented in the knowledge base. Edge types define the different kinds of relations between the entities. This way the knowledge base is designed to be flexible to hold information types defined by the application domain. The intention behind this representation method is to define a framework to represent the information and provide calculation methods, guidelines and methodology for concrete applications. Edge weights (w_E) let the application assign weights to relations. To provide an example, the strength of a friendship relation (close friends, distant friends) in a concrete application scenario can be represented utilizing edge weights. Another example is item similarity, which also can be represented as the weights of the edges. Attributes have been intro-

duced to let applications assign additional information to nodes and edges. Such an item of information represented as attribute value is for instance the rating value of an edge which represents user rating value over a specific item.

3.2 MovieLens. As the method introduced in Section 4.1 is capable to incorporate heterogeneous information source types, we decided to use the MovieLens [8] 1M dataset, which we have found relatively rich in user and item attributes. MovieLens 1M is published by Grouplens[1]. The rating data also contain a time stamp, which is important for the evaluation of our methods. The dataset contains 1 000 209 anonymous ratings of 3 883 movies made by 6 040 MovieLens users who joined MovieLens in 2 000.

MovieLens 1M contains three data files in a proprietary tabular format. The data files hold user, item and rating data. The user data file contains user attributes about each user, as user id, gender, age, occupation and zip-code. The item data file contains item attributes about each movie, as movie id, title and list of genres. The ratings data file contains the user ratings on items, as user id, movie id, rating and time stamp.

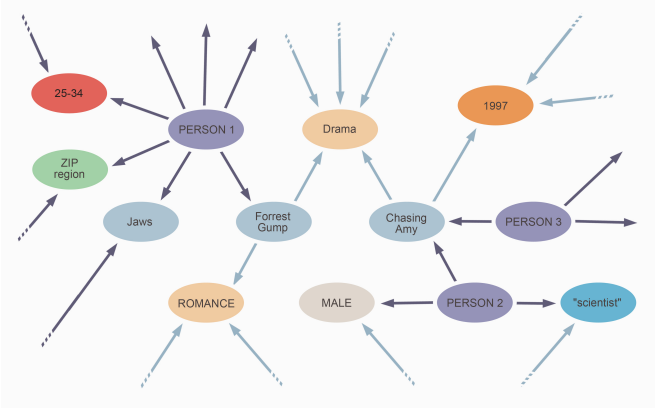

Figure 1: A detailed view of the MovieLens database represented in a directed graph

The MovieLens 1M data is represented with the knowledge base introduced in Section 3.1. As illustrated on Figure 1, the following node types are introduced to represent entities `Person`, `Item`, `AgeCategory`, `Gender`, `Occupation`, `ZipCodeRegion`, `Genre` and `YearOfPublishing`. Nodes of type `Person` represent users, nodes of type `Item` represent movies. User attribute values are represented with nodes. Node type `AgeCategory` is used to represent age category, `Gender` to represent gender, `Occupation` to represent occupation and `ZipCodeRegion` to represent U.S. region. The type `ZipCodeRegion` is a calculated attribute value. The original dataset contains U.S. zip codes. The first digit of the U.S. zip code system represents postal region, which information has been encoded into the attribute nodes. Item attributes are represented similarly. Nodes of type `Genre` represent item genre, nodes of type `YearOfPublishing` represent different years when particular movies were published.

The following edge types have been introduced to represent relations between the entities in the knowledge base `PersonAgeCategory`, `PersonGender`, `PersonOccupation`, `PersonZipCodeRegion`. `ItemGenre`, `ItemYearOfPublishing` and `ItemRating`. Edge types starting with `Person` represent relations between users and user attribute nodes representing user attribute values. For instance edges of type `PersonAgeCategory` represent the information revealing that a person belongs to an age category, `PersonGender` represents the gender of a person, `PersonOccupation` represents the occupation of a person, `PersonZipCodeRegion` represents the place in U.S. region where a person lives. Edge types starting with `Item` except `ItemRating` represent relations between items and nodes representing item attribute values. Edges of type `ItemGenre` represent the information showing which specific genre a movie belongs to, `ItemYearOfPublishing` represents that a movie was published in a specific year.

Edges of type `ItemRating` existing between users and items represent that a user rated an item with a specific value of rating. In this case the rating value was assigned to the edge as an edge attribute. The edges of this type have a special, additional attribute, the rating value. During the import process the rating values are transformed to the $[0.2, 1]$ real interval from the $[1, 5]$ integer interval by dividing the rating values by 5.

4 Recommendation.

Cold start is a common and frequently mentioned problem of recommender systems. Collaborative filtering has no reliable information to derive recommendations from for a newcoming user because the user did not express interest in a sufficient number of items. Content based methods also need information to be able to model the taste of the user. Based on this information, relevant items can be recommended. Knowledge based methods are a possible solution for this problem but their drawback is that these methods in most cases also require user interaction. Our objective is to start recommending relevant items as early as possible. To accomplish this, the highest possible amount of information is rep-

[1] `http://www.grouplens.org/`

resented and a calculation method is defined, which has the ability to derive useful information from heterogeneous data. This strategy ensures high coverage. We define the coverage as the percentage of the cases the recommendation method was able to provide a rating estimation until the corresponding step.

We introduce a spreading activation [15] based technique, which – because of the similar technique – we call recommendation spreading. Spreading activation is a well-known method in the field of semantic networks, neural networks and associative networks [1]. By utilizing spreading activation, our calculation method is able to combine different paths between the source node (i.e. the person we are generating recommendations for) and the recommended nodes (i.e. the items we are recommending). The method is sensitive to the length of each path, in order to downgrade the influence of nodes topographically far from the node in question, i.e. the node the recommendations are generated for.

4.1 Spreading. Recommendation spreading is an iterative method. In the initial step, the activation of the source nodes, is set to a constant value, in most cases to 1. The source nodes are the nodes the recommendations are generated for. In the simplest case the set of source nodes consists of one node, the node representing a person to generate the recommendations for.[2] In each step for each node a part of the activation is distributed to the neighbouring nodes, another part is kept at the activating node. The former parameter, which determines the amount of distributed activation is called `spreading relax`. The latter parameter, which determines the amount of activation kept at the node is named `activation relax`. Both parameters are real numbers and are global settings for the whole network. The outgoing activation is divided along the outgoing edges based on the weight of the edges. All these spreading values are summed up at each receiving node. This way the sum of the activations received by the destination nodes is the same amount as the activations distributed from the source node. If it's important to keep the sum of the activations at a constant level in the network, the sum of activation relax and spreading relax must be equal to one.[3] The concrete spreading relax and activation relax values are the parameters of our spreading method.

Another feature of our spreading methods is the threshold constant. If the activation of a node falls below the threshold constant, its activation will be set to zero [2]. The threshold is introduced to avoid unnecessary computation of activations close to zero. This parameter is called `activation threshold`.

There are various options to define the termination criteria for recommendation spreading. A relatively simple solution is the one based on iteration step, i.e. to stop the iteration after a certain iteration step count is reached. A delta based termination criterion could also be used meaning to run the iteration until there is no significant change in the activations [17]. We decided to use a step limit, because we think that this termination criterion fits better to our intended purpose. In a recommendation scenario a method is necessary, which delivers nodes topographically close to the source nodes. It is also important to mention that calculating a delta based termination criterion is resource intensive, while step limit is cheap to compute. A limit on iteration steps has been introduced, which parameter is called `step limit`.

4.2 Network Based Comparison. In the following we are comparing collaborative filtering with our approach.

Figure 2 illustrates a simplified collaborative filtering scenario. Nodes labeled with "U" represent users, nodes labeled with "I" indicate items. Edges labeled with "R" represent ratings. In this sample scenario we are generating rating estimation for user "U1". User "U1" rated two items, "I1" and "I2" in common with user "U2". User "U1" rated item "I3" in common with user "U3". The rating estimation for item "I4" is calculated as a weighted sum of rating "R7" and "R8". The weights for rating "R7" are influenced by dashed edges, namely "R1", "R2", "R4" and "R5", the weights for rating "R8" are influenced by dotted edges, "R3" and "R6".

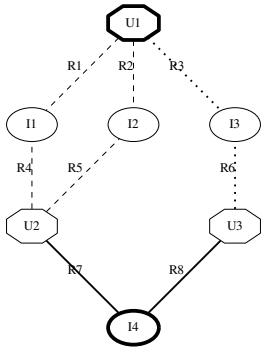

Figure 2: Collaborative Filtering

Figure 3 illustrates our approach, where we show how heterogeneous information can be incorporated to

[2]It is possible to start the recommendation spreading from multiple nodes. For example if a user is browsing an item, then the spreading can be started from the two nodes representing the user and the item. This way the final recommendation result can be influenced by both the user and the browsed item.

[3]Setting the weights to appropriate values won't prevent spreading methods to lose activation. In a case of a node with no outgoing edges the node cannot redistribute its activation to any neighbouring node, thus its activation will be lost. A possible solution to overcome this problem is to bind all nodes to the source node with a backlink edge. This method has other advantages as discussed by Zielger et al. [17], called avoidance of dead ends.

produce rating estimation. Nodes labeled with "U" represent users, nodes labeled with "I" indicate items, the node labeled with "A" represents age category. Age category is an example how to represent user attributes. Item attributes can be represented similarly. Edges labeled with "R" represent ratings, edges labeled with "A" mean user belongs to an age category, edge labeled with "S" means item similarity. User "U1" rated item "I1" and "I2" in common with user "U2". The dashed edges (R1, R2, R4, R5) represent the influence of user similarity on the weight of rating "R7" on final estimation. User "U1" belongs to the same age group "A" with user "U3". The solid edges (A1, A2) represent the same age group relation and their influence to the weight of rating "R8". It also illustrates how user and item attributes can influence the final recommendation result. The dotted path (R3,S1,R6) starts from user "U1" with a rating edge, continues with an edge representing item similarity and ends with a rating edge. The semantics behind this path could be described as user "U1" and user "U4" rated a similar item. Activation received through the dotted path specifies the weight of rating "R9".

4.3 Rating Weights. The introduced recommendation method can also be treated as the generalization of collaborative filtering in the case of representation of heterogeneous information. While calculating recommendation spreading, the activation flowed through each rating edge is accumulated. We denote this value with a, which variable is indexed with the edge. To calculate the weighted sum of estimated rating, these accumulated values are used as the weights of rating values belonging to the specific edge. Recommendation spreading calculates rating estimations with the following formula

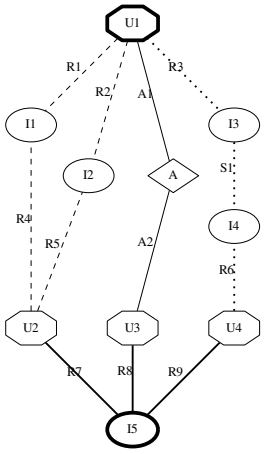

Figure 3: Recommendation Spreading

$$\hat{r}_{u,i} = \bar{r}_u + \frac{\sum_{e \in E_{ItemRating} \wedge i \in e \wedge v \in e \wedge i \neq v} (r_{v,i} - \bar{r_v}) a_e}{\sum_{e \in E_{ItemRating} \wedge i \in e \wedge v \in e \wedge i \neq v} a_e}$$

, where $\hat{r}_{u,i}$ is the estimated rating value for user u on item i, $\bar{r}_u$ and $\bar{r_v}$ denote the average of the already issued ratings by user u and v respectively, $r_{v,i}$ is the known rating value of user v on item i, a_e is the aggregated activation flowed through via edge e, $E_{ItemRating} = \{e \in E | t_E(e) = \texttt{ItemRating}\}$ is the set of edges of type `ItemRating`.

Rating edges are drawn between users and items. This means that in the sample case shown in Figure 3 user similarity between `U1` and `U2` can be defined as the activation arrived from `U1` to `U2` through the network. [4]

4.4 Flow direction. The knowledge base represents the information with a weighted directed graph. Figure 3 shows a sample spreading scenario. If "Item 5" would be recommended to "User 1", a directed path between these nodes couldn't be found. With a recommendation spreading method flowing only in the direction of the edges coverage would be lost. Spreading on an undirected graph also means that if a node received an activation in step i, in step $i+1$ it will spread a part of its activation back to the node it received activation from through the previously receiving edge.

5 Evaluation.

5.1 Evaluation Method. We call our evaluation method time series evaluation, which is an iterative method based on time stamped data. The evaluation iterates through time stamped data of sample items in ascending order, while repeating the following operations

1. take the next rating sample from the database,
2. ask for rating estimation from the recommender engine,
3. calculate the rating error and record it to the evaluation log,
4. add the true rating value to the knowledge base as an edge of type `ItemRating`.

Before starting the evaluation, the knowledge base is filled with all the information found in the MovieLens 1M dataset except the `ItemRating` edges. We decided to use this method to simulate real life scenarios, with a focus on the ability of comparing methods in the beginning, cold start steps, when there is only a low amount of rating information available in the knowledge base. The knowledge base is filled with additional information (`ItemRating` edges) during the evaluation process.

5.2 Numerical Experiment. We were interested in comparing different `step limit` settings to collaborative filtering. Table 1 summarizes the methods which were evaluated in the experiment. For easier reference

[4] The statement holds, because `U2` has only one outgoing rating edge. The activation leaves from `U2` only via `R7`.

Name	Method	Method parameters
CF	Collaborative Filtering	–
S3	Recommendation Spreading	Step limit: 3
S4	Recommendation Spreading	Step limit: 4
S5	Recommendation Spreading	Step limit: 5
S6	Recommendation Spreading	Step limit: 6
S7	Recommendation Spreading	Step limit: 7
S8	Recommendation Spreading	Step limit: 8

Table 1: Engines and configurations

later we assigned a name to these methods which can be found in the first column of the table. We evaluated the recommendation spreading method on the MovieLens 1M dataset represented as described in Section 3.2. Time series evaluation method was conducted as described in Section 5.1. As we were interested in how the methods perform in the information sparse environment, we ran the experiment on the first 10 002 rating values of the sample dataset. The benchmark method in the experiments is collaborative filtering with a Pearson correlation based similarity calculation method. The `activation relax` parameter of the spreading method has been set to 0.5, the `spreading relax` has been also set to 0.5. The `activation threshold` of spreading methods has been set to 0, meaning no thresholding, in order to see the pure, unoptimized performance of spreading methods.

5.3 Rating Estimation Error. Figure 4 compares two methods, namely S3 and CF. We decided to compare S3 with CF as S3 is the most restricted spreading method. Spreading methods running for higher step limits have a higher chance to deliver recommendation estimations of better quality. The horizontal axis of the figure represents evaluation steps, the vertical axis represents the MAE until the corresponding step. Seeing the step interval below 1 000, the figure shows that the recommendation spreading method has a higher coverage in the cold start case. Furthermore, in this region the quality of the estimated values of S3 are higher than the estimated values of the CF method, which statement holds until approximately the 3 000th step. To summarize the information on Figure 4, recommendation spreading converges faster and it has a higher coverage in the cold start case. Coverage is defined as the percentage of cases the recommendation method was able to provide a rating estimation at until the corresponding evaluation step.

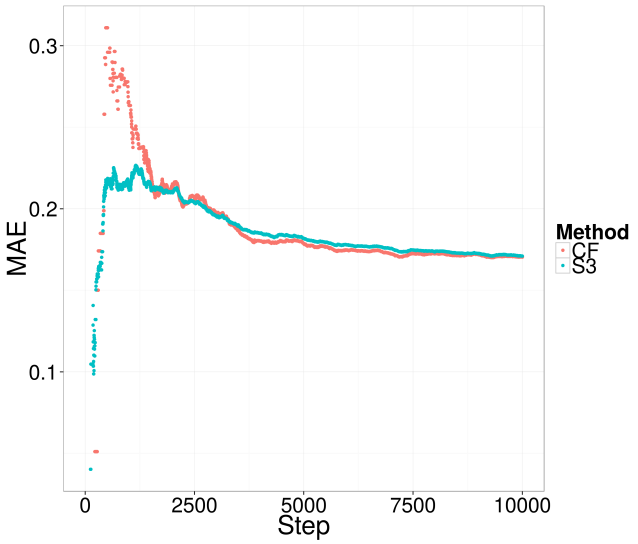

Figure 4: Comparing the MAE of recommendation spreading and collaborative filtering

5.4 Coverage. Figure 5 compares the coverage of S3 and CF methods. The horizontal axis of the figure represents evaluation steps, the vertical axis represents the coverage at the corresponding step. Figure 5 shows that recommendation spreading provides higher coverage than collaborative filtering. The difference on coverage is also high in the beginning steps, when the knowledge graph is more sparse on true rating values. It means that recommendation spreading performs better in the cold start case than collaborative filtering. We explain this by treating the coverage problem as finding a path between two nodes. While collaborative filtering can operate on a restricted set of edges (only on the `ItemRating` edges), recommendation spreading can utilize any type of edge. This is the reason the spreading based method can reach the item node in a higher number of cases.

Engine name	Coverage	MAE
CF	6 118	0,170 4
S3	7 897	0,171 0
S4	7 897	0,171 0
S5	7 910	0,171 1
S6	7 910	0,170 8
S7	7 910	0,170 7
S8	7 910	0,170 5

Table 2: Coverage and MAE of different engines at the last evaluation step

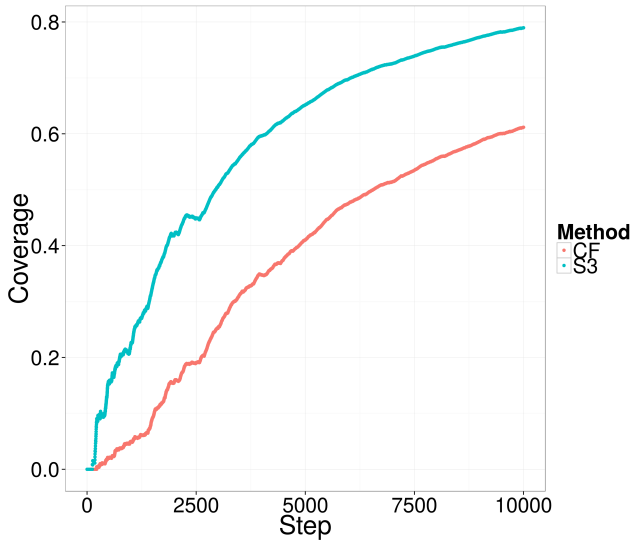

Figure 5: Comparing the coverage of recommendation spreading and collaborative filtering

Table 2 contains the coverage and MAE values of the engines in the experiment at the last evaluation step. The MAE values are very similar, differing only at the third digit. It means that recommender spreading is a method which successfully combines heterogeneous information sources with the same estimation error as collaborative filtering on the MovieLens 1M dataset. Regarding coverage, Table 2 provides two insights. As the spreading method has more options to reach the destination node from the source node, this method is able to estimate ratings in more cases. The second consequence is that a higher spreading `step limit` value does not necessarily lead to a significantly higher coverage. It means that this parameter is sensitive to the underlying data or application domain and should be fine tuned for each dataset or application.

6 Conclusion.

The results show that a rating estimation method was developed which has very similar prediction error as collaborative filtering has. As by its nature recommendation spreading works from a higher number of rating estimations, the method has a higher coverage than the benchmark method has. Comparing recommendation spreading methods configured to different spread `step limits` shows that while the coverage increases with the step limit, the precision of rating estimation does not increase or decrease. It means if an application of the method needs an estimation on a specific item, the iteration can be stopped as spreading reached the node representing the item, because further spreading does not increase the precision.

Next to its higher coverage, an important property of recommendation spreading is its faster convergence. The results can be explained by a higher number of rating values reached to aggregate. As the results also show that the higher the number of aggregated rating values leads to the lower error of rating estimation, the faster convergence can be the consequence of the aggregation of a higher number of rating values also in the beginning, cold start case. It was also shown that in the long term, the introduced method has very similar precision as collaborative filtering has. It means that a method was developed which is able to combine a higher number of ratings while not increasing the error of rating estimation.

We would like to extend the knowledge base or the recommendation engines with an additional information, the information type weight function. Type weights express the strength of a relation type. This information can be used by the calculation method. Introducing type weights, our intention is to let the model or the calculation method store the importance of different relation types. For example the weight of the relation type representing friendship can be set to 0.8, the weight of the relation type representing country of production can be set to 0.4. The values express the importance of the various information types. The calculation methods can use this information when calculating recommendations, for instance by multiplying the relation type weight with the relation weight. The potential of introducing type weights can be found in the learning capability of the recommendation methods. If a calculation method is capable to react on user feedback, it can have a training method to adjust relation type weights according to feedbacks from the environment. Tuning these weights can lead to an increase in the recommendation quality. Manual tuning requires a domain expert and does not guarantee a quick and better result. One of our future plans is to develop a training method which adjusts the weights of the different information source types based on user feedbacks, letting the recommendation method continuously adapt to the changes in the environment.

Currently the building blocks of the knowledge base are information representation units. The graph based model represents the information with nodes and edges. To utilize the information collected in the knowledge base, recommendation spreading algorithm has been used. There are several options to process this information, for example we could also work with a random walk based method. In Section 5 we showed that recommendation spreading has the potential to have a high

coverage but the error of the rating estimation could not have been made lower. Our suspicion is that by introducing a finer grain method, it would be possible to increase the recommendation quality. One option is to utilize neural networks, thus to change information representation units – graph nodes – to artificial neurons to let the network adapt to its environment by training itself.

References

[1] Nicholas V. Findler, editor. *Associative Networks: The Representation and Use of Knowledge of Computers*. Academic Pr, 1979.

[2] Stephan Gouws, GJ Van Rooyen, and Herman A Engelbrecht. Measuring conceptual similarity by spreading activation over Wikipedia's hyperlink structure. In *Proceedings of the 2nd Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources*, page 46, 2010.

[3] Mark Granovetter. The Strength of Weak Ties. *The American Journal of Sociology*, 78(6):1360–1380, 1973.

[4] Hansu Gu, Mike Gartrell, Liang Zhang, Qin Lv, and Dirk Grunwald. AnchorMF: Towards Effective Event Context Identification. In *Proceedings of the 22Nd ACM International Conference on Conference on Information & Knowledge Management*, CIKM '13, pages 629–638, New York, NY, USA, 2013. ACM.

[5] R. Guha, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. Propagation of trust and distrust. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 403–412, New York, NY, USA, 2004. ACM.

[6] Ido Guy, Naama Zwerdling, David Carmel, Inbal Ronen, Erel Uziel, Sivan Yogev, and Shila Ofek-Koifman. Personalized recommendation of social software items based on social relations. In Lawrence D. Bergman, Alexander Tuzhilin, Robin D. Burke, Alexander Felfernig, and Lars Schmidt-Thieme, editors, *RecSys*, pages 53–60. ACM, 2009.

[7] Jianming He. *A Social Network-based Recommender System*. PhD thesis, Los Angeles, CA, USA, 2010. AAI3437557.

[8] J.L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl. An Algorithmic Framework for Performing Collaborative Filtering. In *Proceedings of International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 230–237, Berkeley, USA, August 1999.

[9] Balázs Hidasi and Domonkos Tikk. Fast ALS-Based Tensor Factorization for Context-Aware Recommendation from Implicit Feedback. In Peter A. Flach, Tijl De Bie, and Nello Cristianini, editors, *ECML/PKDD (2)*, volume 7524 of *Lecture Notes in Computer Science*, pages 67–82. Springer, 2012.

[10] Tim Hussein and Sebastian Neuhaus. Explanation of Spreading Activation Based Recommendations. In *Proceedings of the 1st International Workshop on Semantic Models for Adaptive Interactive Systems*, SEMAIS '10, pages 24–28, New York, NY, USA, 2010. ACM.

[11] Audun Jøsang, Stephen Marsh, and Simon Pope. Exploring Different Types of Trust Propagation. In Ketil Stølen, William H. Winsborough, Fabio Martinelli, and Fabio Massacci, editors, *iTrust*, volume 3986 of *Lecture Notes in Computer Science*, pages 179–192. Springer, 2006.

[12] P. Kazienko, K. Musial, and T. Kajdanowicz. Multidimensional Social Network in the Social Recommender System. *Trans. Sys. Man Cyber. Part A*, 41(4):746–759, July 2011.

[13] Ioannis Konstas, Vassilios Stathopoulos, and Joemon M. Jose. On Social Networks and Collaborative Recommendation. In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 195–202, New York, NY, USA, 2009. ACM.

[14] Paolo Massa and Paolo Avesani. Trust-Aware Collaborative Filtering for Recommender Systems. In Robert Meersman and Zahir Tari, editors, *CoopIS/DOA/ODBASE (1)*, volume 3290 of *Lecture Notes in Computer Science*, pages 492–508. Springer, 2004.

[15] M. Ross Quillian. Semantic memory. In Marvin Minsky, editor, *Semantic Information Processing*, pages 227–270. MIT Press, Cambridge, MA, 1968.

[16] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW '94, pages 175–186, New York, NY, USA, 1994. ACM.

[17] Cai-Nicolas Ziegler and Georg Lausen. Propagation Models for Trust and Distrust in Social Networks. *Information Systems Frontiers*, 7(4-5):337–358, December 2005.