

Explanation of spreading activation based recommendations

Tim Hussein and Sebastian Neuhaus

University of Duisburg-Essen

Lotharstr. 65, 47057 Duisburg, Germany

tim.hussein@uni-due.de, sebastian.neuhaus@stud.uni-due.de

ABSTRACT

In this paper, we introduce an approach for explaining recommendations in environments that are based on semantic models. Using a constrained Spreading Activation (CSA) technique for recommendation generation, we store and exploit the activation paths leading to recommendations. These paths later can be used to generate both verbal explanations and relevance feedback forms.

Author Keywords

Model-driven Recommendations, Explanations, Visualization, Traceability, Scrutability, Relevance Feedback

ACM Classification Keywords

D.1.2 Information Systems: Information Storage and Retrieval—*Information Search and Retrieval*

INTRODUCTION

Online shops as well as digital libraries or news portals offer a vast amount of available information. Although this variety of choice is generally a benefit for the user, the flood of information usually distracts the users.

This variety of choice may be too much for the user and, if not filtered adequately, may cause him to leave a website. The problem is not the amount of data, but an adequate way of filtering *the right* content at the right time, so that the user finds exactly those items he or she is looking for.

One of the most successful ways to support the user's selection process is the use of so called *Recommender Systems* to hide irrelevant information and present only items that the user is supposed to like. The decision is usually based on the user's past purchases, ratings, or visits as well as his preferences and needs (if he has decided to name them explicitly). As a result, recommender systems show personalized information of products the user a) has not seen yet and b) is likely to be interested in. In doing so, they facilitate the exploration problem.

The field of recommender systems is a vivid research area since the 1990s. There have been various approaches to enhance the user interface by recommending items or functionalities the user is supposed to be interested in. Recommender systems have been established as an independent research area during the 1990s, having their roots in various disciplines like cognitive science and information retrieval. Most techniques can be roughly divided into content-based [11], collaboration-based [13], and hybrid approaches [4, 3].

Content-based systems incorporate features associated to the objects of interest. User ratings or transactions can be analyzed in order to find out his or her interests, to recommend items similar to those bought in the past or rated as positive. For instance, neural nets, decision trees and vector-based representations can be used for that purpose. Collaborative filtering (CF) methods are supposed to be the most widely implemented recommendation techniques. They can be partitioned into classical User-based-CF and Item-based CF methods. User-based techniques identify so-called mentors for a user by generating vectors from the user's ratings and comparing those vectors, for instance, by correlation measurement or cosine. They assume that similar users are interested in similar items. So the mentors ratings for items are multiplied by the mentors similarity in order to predict the recommendations.

While research mostly focuses on improving algorithms, traceability is often neglected, although trust and user acceptance are crucial for these kinds of systems [8]. Some recommender systems are based on semantic models and techniques like Spreading Activation. In this paper, we illustrate a way of explaining such recommendations and allowing the user to tweak the recommendation process easily.

ADAPTATION BY CONSTRAINED SPREADING ACTIVATION (CSA)

In 1975, Collins & Loftus introduced a technique called *Spreading Activation* [6]. This model was originally applied in the fields of psycho linguistics and semantic priming [1]. Later, the idea was adopted by computer scientists: Spreading activation techniques have successfully been used in several research areas in computer science, most notably in information retrieval [5, 7, 2]. The principles of spreading activation have also been used by Pirolli & Card [12] in their information foraging theory.

The basic concept behind Spreading Activation is that all relevant information is mapped on a graph as nodes with a certain “activation level”. Relations between two concepts are represented by a link between the corresponding nodes. If for any reason one or more nodes are activated their activation level arises and the activation is spread to the adjacent nodes (and the ones related to them and so on) like water running through a river bed. Thereby the flow of activation is attenuated the more it strides away from the initially activated node(s). At the end several nodes are activated to a certain degree that are semantically related to the concepts originally selected. We make use of this technique to identify content to be displayed in adaptive areas of a web application [10].

At the beginning, each node has an initial activation value of 0. When the user clicks on a certain item (for instance a DVD), the node representing this item is used as initial node to trigger the activation flow. As a result concepts and items that are related to the current one have a high activation value. A little example should illustrate this: The user might be interested in the movie “Big Fish” and clicks on the corresponding link.

As a result, the system injects an initial activation (for instance 0.5) into the network node representing “Big Fish”. This node then spreads some of the activation to semantically related nodes, like “Ewan McGregor” (who plays the main character of the movie) and “Tim Burton” (who is the director). Both nodes, in turn, spread a certain amount of the received activation to other nodes as well. Ewan McGregor, for instance, activates “Star Wars - Episode I”, and so on. Finally all nodes have a certain activation value that represents their degree of relevance in the current situation. The basic idea is illustrated in Figure 1.

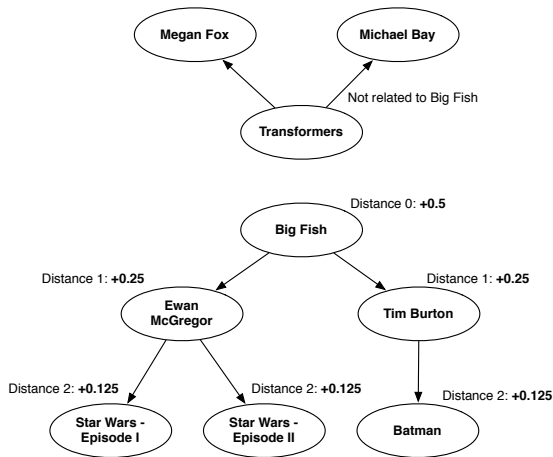


Figure 1. The basic idea behind spreading activation.

After that run the activation values are not reset but refined with every user action. If he clicks on a certain domain item - for instance a concert - this interaction is taken into account, too. The node representing that particular concert now is the initial node to another Spreading Activation run and transmit

activation energy to all related concepts and items. That may include the concept of a concert in general (and thereby activating other concerts), the artist, the music genre, the venue of the concert and so on. Each interaction adopts the weights more and more to the current context and the user’s interests.

At this point no adaptation is performed yet. Only the underlying models are adjusted to the current context and usage behavior. We believe that it is a good idea to separate these model adjustment mechanisms from the process of web page generation. Any changes in the page generation and presentation framework won’t affect the reasoning part of the system and vice versa.

Several algorithms have been developed to implement the concept of spreading activation. Details and a comparison can be found in Huang et al. [9]. We chose the so called *Branch-and-bound* approach.

The branch-and-bound algorithm

The following steps describe a single Spreading Activation run. As mentioned earlier the activation values are not reset after each cycle so that the networks can develop into a kind of user and context profile. During a Spreading Activation run two phases can be distinguished: Initialization and execution.

Initialization:

Before the actual execution of spreading activation begins, the network must be initialized:

1. The weights for the links are set based on the user’s individual context model. Moreover, in our approach, the network is not necessarily in a blank state when a spreading activation run starts. Therefore, initial activation levels for each node in the network are set. These are based on the resulting activation levels of the previous run.
2. The initial nodes are activated with a certain value. The activation received by the start nodes is added to their previous state. Optionally the new activation level is calculated by applying an activation function to this sum.
3. The initial nodes are inserted into a priority queue ordered by descending activation.

Execution:

After initialization, the following steps are repeated until a defined termination condition is fulfilled or the priority queue is empty. The termination condition can be configured freely, but two pre-defined termination conditions are provided: (1) A maximum of *activated* nodes is reached, (2) a maximum of *processed* nodes is reached. A processed node is a node that has itself propagated activation to adjacent nodes.

1. The node with the highest weight is removed from the queue.
2. The activation of that node is passed on to all adjacent nodes, if this is not prevented by some restriction imposed

on the spreading of activation. If a node j receives activation from an adjacent node i , a new activation level is computed for j .

$$A_j(t+1) = A_j(t) + O_i(t) \times w_{ij} \times a$$

where $A_j(t)$ is the previous activation of j , $O_i(t)$ is the output activation of i at the time t , w_{ij} is the weight of the relation between i and j and a is an attenuation factor. The output activation of a node is the activation it has received. An arbitrary function can be used to keep the values in a predefined range. In most cases a linear or parabolic function will be meaningful.

3. The adjacent nodes that have received activation are inserted into the priority queue unless they have already been marked as processed.
4. The node that passed on its activation to the neighboring nodes is marked as processed.

When a new spreading activation run is triggered the values are not reset, so that the network is refined every time the process is executed. We implemented an aging mechanism by attenuating each activation value by 5% before each new spreading activation run.

TRACEABILITY THROUGH EXPLANATIONS

In adaptive environments, scrutability is an important issue [8].

If humans recommend items to each other, this word-of-mouth recommendations always include a “social context” both are aware of. Somebody wondering which book or movie to buy next would typically ask friends, who are interested in similar genres or have a similar taste. After receiving a recommendation, he has several cognitive options to deal with it in order to find out, if he should trust the recommendation.

He could consider a) the similarity to the recommending person in terms of taste, b) the success rate of previous recommendations given by that person, or c) he could ask the person the explain why he chose that particular recommendation over possible others [14].

These background information regarding a recommendation, or explanations respectively, are usually neglected in electronic recommender systems. Particularly commercial systems hide a lot of information about the recommendation process from the user, which makes the whole process very inscrutable. Of course, sometimes this behavior is necessary in terms of business secrets. But as a result, most recommender systems act like black boxes, leaving the user with only to options: Trust the system – or not.

In most cases, he does not have the opportunity to scrutinize the process. Nor does he know upon which data the system generated the recommendation. That makes it hard for him to know whether he should trust the system or not. Especially considered the fact, that even the best recommender

system may fail from time to time, explanations would be a great benefit for the user.

Our goal was to explain adaptive system behavior to the user. In this paper, we concentrate on explaining recommendations based on CSA as described above. Whenever a user clicks on an item (for instance a DVD), a CSA run is triggered, activating other items on various paths. Our system keeps track of these paths and generates explanations upon them (Figure 2).



Figure 2. Brief explanation of a recommendation.

Basically, the items are displayed that triggered a CSA run resulting in the recommendation. To be more precise, the activation paths leading to the recommendations are analyzed and the starting points of those paths are displayed.

More detailed explanations are available as well: In addition to regular labels, we enriched all properties in the semantic model with strings representing singular and plural verbalizations of them (for instance “is performed” and “are performed”). Using relatively simple graph algorithms, we can make use of these verbalizations to generate more sophisticated explanations like those illustrated in Figure 3.

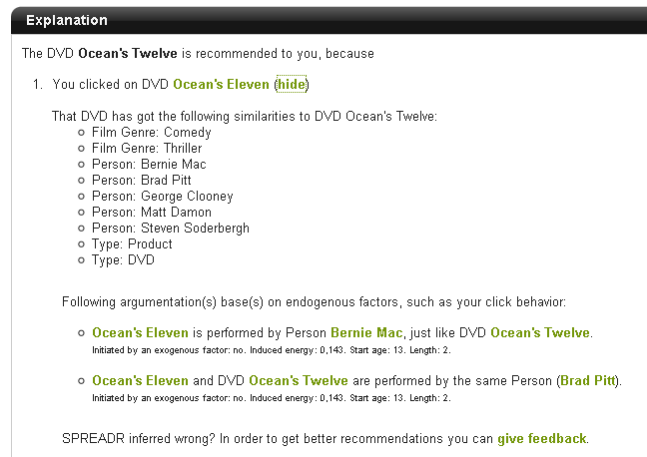


Figure 3. Detailed explanation of a recommendation.

Without the verbalization, an activation path would look like this:

CD Phil Collins - Face value → hasParticipant → Phil Collins → participatesIn → CD Phil Collins - Dance into the light → hasGenre → Pop → isGenreOf CD

Genesis - Calling all stations → hasParticipant → Genesis → participatesIn → CD Genesis - Foxtrot.

The verbalization mechanism now looks for certain patterns and generates a human-readable (and understandable) verbal representation of this path.

RELEVANCE FEEDBACK

Sometimes in information retrieval or recommender systems, a presented information does not correspond to the actual user's needs. As a consequence, the user is not satisfied with the results. Relevance feedback is a way of supporting the system by giving feedback of the quality of the generated recommendation.

Traditional relevance feedback enhances a retrieval process by incrementally correcting the results. The basic idea is to know more about the user's actual needs. The user has to provide implicit or explicit feedback by answering certain questions. In many cases, these questions are rather vague, because users are not able to name their needs precisely. So, the system tries to narrow the alternatives by iteratively refining the results.

In our prototype, we implemented relevance feedback by automatically generating feedback forms based on the user's surfing behavior. The user can use these forms to tell the system more about his or her actual interests to avoid wrong recommendations.

Figure 4 illustrates a form for relevance feedback. The user has the option to react on the recommendations, since the system is able to automatically generate feedback forms by exploiting the activation paths. If he states that he does not want the system to exploit certain items or classes of items, this has an effect on future CSA runs: In case of a negative feedback for instance the item is put on a *stop list*, so that activation energy will not spread through this particular item.

More sophisticated feedback is possible as well, if the user decides that a certain item is generally ok (for instance *Arnold Schwarzenegger*), but not in connection with other items (for instance *comedies*). As a result, action movies by Arnold Schwarzenegger would be recommended as well as books about his career as a bodybuilder or politician – but none of his comedy movies.

In this case, these items, types of items, relations or types of relations are put on the stop list to prevent the spreading activation algorithms from flowing through them.

SUMMARY

In this paper, we presented a way of explaining recommendations based on semantic models and constrained spreading activation. We exploit the activation paths occurring during the recommendation process and use simple graph algorithms to generate complex explanations. In a similar way, relevance feedback forms can automatically be generated to improve future recommendations.

Feedback

Following argumentation(s) base(s) on endogenous factors, such as your click behavior:

- DVD **Terminator** belongs to Film Genre **Action**, just like DVD **Out of sight**.
Initiated by an exogenous factor: no. Induced energy: 0.072. Start age: 16. Length: 2.

If you **don't like/care about** one or more argumentation step(s), please check the corresponding item(s). To submit your choice please click the "Give feedback" button below:

Items	Types of Items	Relations	Types of Relations
I don't like/care about...			
<input type="checkbox"/> the DVD Terminator.			
<input type="checkbox"/> the Film Genre Action.			
<input type="checkbox"/> the DVD Out of sight.			
<input type="button" value="Give feedback"/>			

Figure 4. A form for relevance feedback.

ACKNOWLEDGEMENTS

The research presented in this paper is part of the CON-TICI project, in which the Universities of Duisburg-Essen, Siegen, Hagen, and Aachen take part. CONTICI is funded by the German Research Foundation (Deutsche Forschungsgemeinschaft).

The authors thank Werner Gaulke and Timm Linder for fruitful discussions and implementation support.

REFERENCES

1. J. R. Anderson. A spreading activation theory of memory. *Journal of Verbal Learning and Verbal Behavior*, 22:261–295, 1983.
2. H. Berger, M. Dittenbach, and D. Merkl. An adaptive information retrieval system based on associative networks. In *APCCM '04: Proceedings of the first Asian-Pacific conference on Conceptual modelling*, pages 27–36, Darlinghurst, Australia, Australia, 2004. Australian Computer Society, Inc.
3. R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.
4. M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR Workshop on Recommender Systems*. ACM, 1999.
5. P. R. Cohen and R. Kjeldsen. Information retrieval by constrained spreading activation in semantic networks. *Information Processing and Management*, 23(4):255–268, 1987.
6. A. M. Collins and E. F. Loftus. A spreading activation theory of semantic processing. *Psychological Review*, 82(6):407–428, 1975.
7. F. Crestani. Application of spreading activation techniques in information retrieval. *Artificial Intelligence Review*, 11(6):453–482, 1997.

8. M. Czarkowski and J. Kay. A scrutable adaptive hypertext. In *Proceedings of the Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH '02)*, pages 384–387, London, UK, 2002. Springer.
9. Z. Huang, H. Chen, and D. Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems*, 22(1):116–142, 2004.
10. T. Hussein and J. Ziegler. Adapting web sites by spreading activation in ontologies. In L. Bergman, Kim, Jihie, B. Mobasher, S. Rueger, S. Siersdorfer, S. Sizov, and M. Stolze, editors, *Proceedings of International Workshop on Recommendation and Collaboration*, New York, USA, 2008. ACM.
11. R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204, New York, NY, USA, 2000. ACM.
12. P. Pirolli and S. Card. Information foraging in information access environments. In I. R. Katz, R. Mack, L. Marks, M. B. Rosson, and J. Nielsen, editors, *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 51–58, Denver, Colorado, USA, 1995. ACM Press.
13. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186, New York, NY, USA, 1994. ACM.
14. R. Sinha and K. Swearingen. The role of transparency in recommender systems. In *Extended abstracts on Human factors in computing systems. Part of Chi '02*, pages 830–831, New York, NY, USA, 2002. ACM.