Dynamic Generation of Personalized Hybrid Recommender Systems

Simon Dooms
iMinds-Ghent University
G. Crommenlaan 8, box 201, Ghent, Belgium
simon.dooms@intec.ugent.be

ABSTRACT

The problem of information overload has been a relevant and active research topic for the past twenty years. Since then, numerous algorithms and recommendation approaches have been proposed, which gives rise to a new type of problem: recommendation algorithm overload. Although hybrid recommendation techniques, which combine the strengths of individual recommenders, have become well-accepted, the procedure of building and tuning a hybrid recommender is still a tedious and time-consuming process. In our work, we focus on dynamically building personalized hybrid recommender systems on an individual user basis. By means of a dynamic online learning strategy we combine the most appropriate recommendation algorithms for a user based on realtime relevance feedback. Learning effectiveness of genetic algorithms, machine learning techniques and other optimization approaches will be studied in both an offline and online setting.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering, Relevance feedback, User-centered design

General Terms

Algorithms, Design, Human Factors

Keywords

Recommender systems, hybrid recommender system, self-learning

1. INTRODUCTION

Since the introduction of the collaborative filtering principle by Resnick et al. [1] in 1994, numerous algorithms and recommendation approaches have been proposed to tackle the information overload problem. Every one of these algorithms in its own way attempts to connect users to relevant content. Roughly classified, recommendation algorithms are typically divided into categories based on the data they exploit. Collaborative algorithms use rating behavior of the community (i.e., other users or items), content-based algorithms process available content data (e.g., item features) and knowledge-based algorithms exploit general knowledge available in the domain of the recommendation use case.

In recent years however, rather than focussing on mathematically improving classic algorithms to squeeze out every last bit of recommendation accuracy, new types of algorithms are on the rise. New trends as social media and smartphones are increasingly providing new and complexer types of user data like social data (e.g., friend connections) and context (e.g., location, time of day, mood, etc.), which results in new categories of recommendation algorithms that specifically target these kinds of data.

While recommendation algorithms used to be competing against each other, nowadays it has been generally accepted that every algorithm has its own focus, optimal use cases, advantages and disadvantages and that hybrid recommenders, which combine multiple recommenders, are able to further increase recommendation quality and overcome individual drawbacks [2, 3, 4]. Although hybrid recommenders are well-accepted and have shown their merits in multiple competitions (such as the Netflix prize), the procedure of actually building and tweaking the hybrid recommender is still a tedious and time-consuming process.

In our work, we focus on building a future-proof hybrid recommendation platform that can seamlessly integrate existing recommendation algorithms and dynamically combine them into a best-fit hybrid recommender. While most mainstream recommendation algorithms typically start from the notion that users in a system behave similarly [5], recent research is turning more towards the idea that every user is unique and (combinations of) different algorithms may be best for different users [4, 6, 7]. Therefore, every user would probably benefit from a uniquely tailored hybrid recommender system. A self-learning hybrid platform could prove to be of great value for e.g., industrial use cases where often different algorithms are experimented with simultaneously, and both users and industry would benefit from the improved user experience.

We define the following research questions:

- Do (all) users benefit from personalized hybrid recommender systems?
- How can a hybrid recommender be dynamically adjusted?

- What kind of artificial intelligence techniques can be applied to this problem?
- What kind of evaluation metrics should be considered?
- How can we measure and take into account online user relevance feedback?

To address the stated research questions, the problem of building a hybrid recommender system will be reformulated as an optimization task to which known techniques from the domain of machine learning can then be applied.

2. RELATED WORK

Hybrid recommender systems were first categorized by Burke et al. [2] in function of how they combine individual recommenders (e.g., weighted, cascade, mixed, etc.). Early hybrid recommender systems often internally merged two classical algorithms e.g. a form of collaborative filtering (CF) with content-based filtering (CBF) to cope with specific failing use cases. Cornelis et al. [8] for example combined elements of CF and CBF to recommend time-specific items (i.e., events), which get rated only after users have consumed (i.e., visited) the items. Since in these types of hybrids, specific properties of the individual algorithms are exploited, they lack extendibility and easy integration of more and other types of algorithms.

Hybrid recommenders regarding individual recommendation algorithms as black boxes, are more interesting because they easily allow to extend their models with other algorithms of any type. One famous example of such a system is the winning submission to the Netflix prize. Belkor et al. [9] blended 107 individual recommenders into one weighted hybrid model to achieve an optimal RMSE value. Their optimization target however was overall RMSE, and so the weights for the individual algorithms were the same for every user (i.e., static hybrid).

The AdaRec system [10] proposed a dynamic hybrid strategy that modifies its prediction strategy at runtime to cope with unique domains, trends and user interests. Focus however, is on a switching strategy that selects the most suitable recommendation algorithm rather than composing a dynamic weighted hybrid that incorporates input from all algorithms.

Closely related to this work, is the work of Bellogin et al. [4] where the problem of dynamic weighting of ensemble recommenders was analyzed from an information retrieval (IR) perspective. They proposed adaptations of query performance techniques from ad-hoc IR to define performance predictors in recommender systems. These performance predictors could then be used to dynamically adapt the recommendation strategy to the situation at hand. Our work differs from theirs, in that we do not adapt our system to predicted accuracy, but rather to realtime relevance feedback collected through a user-system interaction process. We furthermore investigate the dynamic ensemble weighting problem from a machine learning perspective and observe a black box approach for individual recommendation algorithms allowing easy integration of numerous algorithms regardless of complexity or type.

3. DYNAMIC FRAMEWORK

To be able to empirically experiment with various strategies and settings, we built a dynamic framework that provides some basic recommendation functionality. Figure 1 visualizes the conceptual layout of our framework. In our research we focus on movies as recommendable items because of the wildly available datasets and easily interpretable domain. To avoid any cold start problems, we start from the MovieLens (1M) dataset but integrate also rating data collected from popular social media platforms to also feature new and relevant items (the most recent movie in the MovieLens 1M dataset is from the year 2000). The rating data is then provided to multiple recommendation algorithms which run in parallel and isolated from each other. Individual algorithms are considered black boxes, and so only their inputs and outputs are being taken into account. Because the system does not rely on any internal properties of the algorithms, new algorithms can easily be 'plugged in'. This approach even allows to add all sorts of hybrid algorithms e.g. cascading algorithms to be integrated without any additional effort (displayed as the '×' algorithm in Figure 1). We currently integrated over 20 algorithms from the recommendation framework MyMediaLite [11] but plan on integrating more from other recommendation frameworks (like LensKit [12], Duine [13] and Mahout[1]) as well.

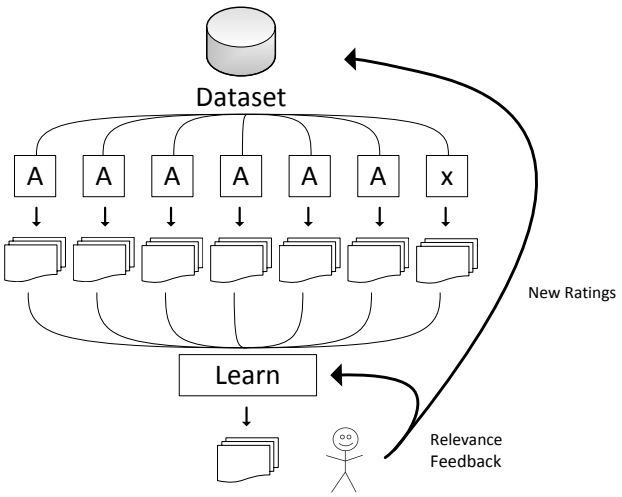

Figure 1: The layout of our dynamic framework. Data is fed to multiple recommendation algorithms in parallel, which are then combined into one hybrid set of recommendations by a self-learning online module that processes user relevance feedback.

The output (i.e., predictions) of the algorithms serve as input to the learning module of the system, which then generates final predictions for a user. Interactively a user is able to provide relevance feedback (i.e., how good are these recommendations?) while providing new ratings. The relevance feedback is propagated back to the learning module which adapts in realtime and new results can be made available to the user. New ratings in their turn are propagated to the backend of the system where they are merged with the original dataset and provide new data for the recommendation algorithms.

4. LEARNING APPROACH

[1]http://mahout.apache.org

The learning approach, or *module* as referred to in the previous section, is the very heart of this research. We want to dynamically combine predictions made by multiple algorithms and use reinforcement learning strategies to adapt in realtime to user feedback. To do so, we will be redefining our problem as an optimization task. We adopt the notation from [14] to define a dynamic ensemble recommender as:

$$g(u,i) = \gamma_{a_1}(u) * g_{a_1}(u,i) + ... + \gamma_{a_n}(u) * g_{a_n}(u,i)$$

where γ is a weighting factor indicating the importance value of the contribution for every algorithm a to the final prediction score g for a user u and item i. We now define an objective function:

$$f(\gamma(u))$$

where

$$\gamma(u) = (\gamma_{a_1}(u), ..., \gamma_{a_n}(u))$$

with n the total number of algorithms. By choosing an objective function f, we can now minimize (or maximize) f by creating an appropriate vector of algorithm weights $\gamma(u)$. Interesting choices for objective functions are for example typical evaluation metrics in the recommendation domain like RMSE, MAE, etc. When the objective function is selected, known optimization techniques can be applied to generate good choices for the weight vector $\gamma(u)$. The implied research challenge for the learning module will be twofold. First, we must define what we want to optimize. Do we want minimal RMSE? Maximum precision or recall? Or even other metrics as 'number of user clicks', as long as they are measurable, can be optimized for. Secondly, an appropriate optimization technique must be selected to generate a good solution in an appropriate time frame. Since we want to update the hybrid recommendations for a user dynamically while user feedback is provided, the appropriate time frame in this context is 'almost realtime'.

We are currently investigating the applicability of genetic algorithms which, according to our earliest experiments, are able to rapidly provide near-optimal solutions to our problem when properly tuned.

Figure 2 shows the results of a small experiment where we test our dynamic hybrid approach on 10 randomly selected users from the MovieLens 1M dataset. For these users, we trained our genetic algorithm to optimize their algorithm weight vectors (optimizing for RMSE) for 25 individual recommendation algorithms. We compared our dynamic hybrid recommender with a baseline switching strategy that for each user selects the best algorithm (i.e., lowest RMSE). While the difference varies from user to user, an overall advantage of our weighted hybrid approach (over the switching approach) can be observed.

When we manually inspect the generated algorithm weight vectors, we find them to be very different for each user and so our early results seem to confirm that it makes sense to personalize these weights on an individual user level.

We plan on expanding our experiments to other machine learning techniques and to do more formal evaluations, first focusing on offline testing, where user data is simulated or *replayed*, but gradually moving to online testing where actual users can become involved in the evaluation process.

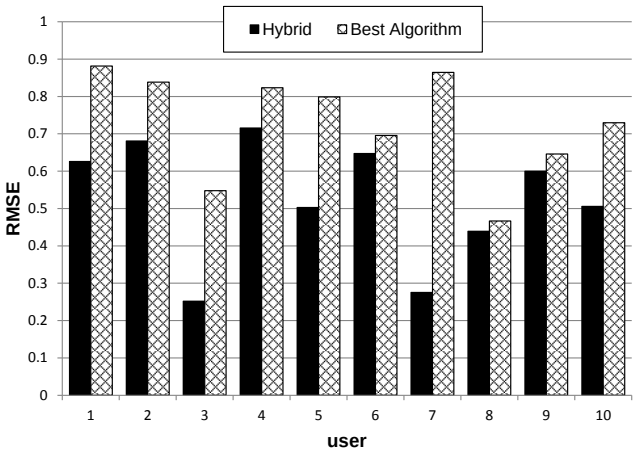

Figure 2: RMSE comparison of our dynamic hybrid recommender, optimized by a genetic algorithm, and a baseline hybrid switching strategy for 10 random MovieLens users.

5. AVOIDING FILTER BUBBLES

In 2011, Eli Pariser caused a stir in the recommender systems domain when he introduced the concept of the *Filter Bubble*[2]. He claimed that, because of personalization, more and more users of online platforms will be trapped in their own separate, filtered *bubbles* of information. Such a bubble would virtually surround users with content tailored to their interests and therefore at the same time also keeps them from thinking outside the 'bubble'. They are no longer able to learn new things, evolve and change interests. A panel in the RecSys 2011 conference discussed the topic, and the take-away message was that personalization is fine as long as users can be given a certain amount of control (e.g., choose to disable personalized results) and recommender systems can be made as transparent as possible (e.g., explain the origin of a recommendation).

In our work, we intend to avoid bubble issues by allowing users to take control of their recommendations. In the context of dynamic ensembles, being in control could mean for example being able to manipulate the weight vector that controls the importance of underlying algorithms. Transparency can also be easily provided by allowing users to inspect their weight vector. Whether these functionalities should be available for all users, or for power users only (e.g., recommendation administrators) may differ, and depends largely on the intended use case.

6. DISCUSSION

We aim for our work to be a usable blue print for real-life recommender system applications. Especially in industry, content providers are not so much interested in what recommendation algorithm is deployed as long as it delivers the expected quality. Therefore, a self-learning system that automatically assemblies good hybrids from individual 'plugged-in' algorithms could be of great value. Furthermore, it allows to introduce new trending algorithms at production time without fundamentally disrupting previous user experience. If newly introduced algorithms turn out to

[2]http://www.thefilterbubble.com/ted-talk

deliver good predictions, their importance weights can gradually increase and thereby also their contribution to the final user recommendations.

Some recommendation algorithms, such as neighborhood-based algorithms, can be run with a variety of different settings such as different neighborhood sizes, different similarity metrics, etc. Our hybrid approach, makes it very easy to empirically test appropriate settings, by plugging in multiple versions of the same algorithm in the system, each time with different settings. The best settings will then automatically prevail.

Scalability issues are avoided in the system by the decoupling of the individual recommendation calculation and the final hybrid visualization of the results to users. While a slow calculation backend can periodically re-run the individual recommendation algorithms in batch, a fast online learning module on the frontend can provide users with an interactive and realtime feeling by instantly adapting to user feedback. Finally, we note that because the vector of algorithm weights can be made manipulable to the user, the system is able to account for some very user-specific situations. By for example including a recommendation algorithm that predicts ratings proportional to how recent movies are, the manipulation of the weight of this algorithm allows users to choose for themselves how much they want the recentness of movies to contribute their recommendations. Another interesting example is the random recommender. By including the random recommender as one of the individual recommendation algorithms, users may introduce an appropriate value of randomness thereby manipulating the serendipity (and diversity) of their recommendations.

7. CONCLUSIONS AND PLANNED WORK

In this work, we wish to solve the problem that emerged from solving the information overload problem: the recommendation algorithm overload problem. Instead of focusing on quality improvement of individual algorithms, we turn towards hybrid recommendation to combine the advantages and surpass the quality of its individual components. We propose a future-proof, dynamic weighted ensemble approach that allows to generate hybrid recommenders on an individual user basis. To aid our research task, we have implemented a basic recommendation platform that integrates numerous and widely differing algorithms and supports a self-learning strategy to dynamically combine them in an optimal way. We are currently experimenting with genetic algorithms, but plan on involving and comparing multiple types of optimizing techniques like neural nets and other machine learning strategies.

In ongoing work we intend to explore an evaluation strategy, both in an offline and online context, to assess the quality of developed methods and eventually address the research questions proposed in this work.

8. ACKNOWLEDGMENTS

The described research activities were funded by a PhD grant to Simon Dooms of the Agency for Innovation by Science and Technology (IWT Vlaanderen).

9. REFERENCES

[1] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proc. 1994 ACM Conf. Computer supported cooperative work*, pages 175–186. ACM, 1994.

[2] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

[3] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):734–749, 2005.

[4] Alejandro Bellogín. *Performance prediction and evaluation in Recommender Systems: An Information Retrieval perspective*. PhD thesis, Universidad Autonoma de Madrid, November 2012.

[5] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. *Recommender systems: an introduction*. Cambridge University Press, 2010.

[6] Michael Ekstrand and John Riedl. When recommenders fail: predicting recommender failure for algorithm selection and combination. In *Proc. 6th ACM Conf. Recommender systems (RecSys 2012)*, pages 233–236. ACM, 2012.

[7] Benjamin Kille and Sahin Albayrak. Modeling difficulty in recommender systems. In *Workshop on Recommendation Utility Evaluation: Beyond RMSE (RUE 2011)*, page 30, 2012.

[8] Chris Cornelis, Xuetao Guo, Jie Lu, and Guanquang Zhang. A fuzzy relational approach to event recommendation. In *Proc. Indian Int. Conf. Artificial Intelligence*, 2005.

[9] Robert M Bell, Yehuda Koren, and Chris Volinsky. The bellkor solution to the netflix prize. *KorBell Team?s Report to Netflix*, 2007.

[10] Fatih Aksel and Aysenur Birtürk. An adaptive hybrid recommender system that learns domain dynamics. In *Int. Workshop on Handling Concept Drift in Adaptive Information Systems: Importance, Challenges and Solutions (HaCDAIS-2010) at the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, page 49, 2010.

[11] Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. MyMediaLite: A free recommender system library. In *Proc. 5th ACM Conf. Recommender Systems (RecSys 2011)*, 2011.

[12] Michael D Ekstrand, Michael Ludwig, Joseph A Konstan, and John T Riedl. Rethinking the recommender research ecosystem: reproducibility, openness, and lenskit. In *Proc. 5th ACM Conf. Recommender systems (RecSys 2011)*, pages 133–140. ACM, 2011.

[13] M. van Setten. *Supporting People in Finding Information: Hybrid Recommender Systems and Goal-Based Structuring*. PhD thesis, Telematica Instituut, Enschede, December 2005.

[14] Alejandro Bellogín. Predicting performance in recommender systems. In *Proc. 5th ACM Conf. Recommender systems (RecSys 2011)*, pages 371–374. ACM, 2011.