PhD Proposal

João Vinagre
LIAAD - INESC Porto LA
joao.m.silva@inescporto.pt

March 9, 2012

Abstract

Collaborative filtering (CF) has been an active subject of research in the field of recommender systems. Given the ever growing importance of the Web in commerce, learning and governance, and the massification of interactive media, recommender systems are currently a hot topic with many challenges to be faced. Typical applications of CF include recommendation of movies, music and Video-On-Demand items. One important issue is to have scalable, yet accurate, algorithms. Another one is related to the dynamic nature of usage data. This project aims to solve important scalability problems in CF and to develop algorithms able to timely detect and adapt to changes in user preferences. We address scalability issues using incremental algorithms, and applying Matrix Factorization and Clustering methods. We deal with temporal effects using state-of-the-art stream mining techniques, detecting changes and updating models accordingly. Our expected result is a set of highly scalable, fully assessed CF algorithms.

1 State of the art

Much work has focused on the scalability of collaborative filtering (CF). The most widely studied techniques are based on matrix factorization (MF) [18, 11]. One popular method that uses MF is Latent Semantic Analysis (LSA) [3], a technique that uses Singular Value Decomposition (SVD) to produce lower rank matrices by approximation to the initial user-item ratings matrix.

Other approaches to scalability are based on clustering models. In [7] it is shown that incremental co-clustering is a viable solution to scalability issues. In [13] a new incremental clustering method based on the concept of representative objects is presented showing considerable scalability improvements.

Another important problem in CF is that it is frequently convenient, in many applications, to maintain up-to-date recommender models. The complexity of generating a model from a large data set can be very high. Therefore, we need efficient mechanisms for frequent model updates. Incremental algorithms try to reduce this complexity by incrementally updating models, instead of regenerating the whole models, every time new data are available. In [17] an incremental CF algorithm that uses SVD is presented and evaluated. Incremental user-based CF has been presented in [16], where user-user similarities are incrementally up-

dated every time new data is available. In [14] incremental algorithms for binary usage data are proposed and evaluated.

Temporal effects in CF have been the subject of recent work. A recommender system that gradually forgets usage data is evaluated in [12], in the context of drifting user preferences. In [4] time-weighted ratings are used in a CF algorithm, favouring the most recent user activity. A CF recommender based on a stream clustering algorithm that learns from evolving streams of usage data is presented in [15]. In [10], a model that is able to separately deal with multiple changing concepts is proposed. In our recent work [19] we add the ability to forget older data using fading factors and sliding windows to algorithms from [14].

In the field of data stream mining [5], several methods have been proposed to provide algorithms with mechanisms to deal with changes in the probability distribution of streaming data, often known as concept drifts. Most of these mechanisms are based on fixed-size or adaptive windows [20, 5, 8, 1]. Other approaches are based on the idea of applying weights to data elements according to their age [12, 9]. In [6] the authors use fading factors to estimate error and detect change in the probability distribution in data from streaming applications. The MOA project [2] provides an extensible framework for data stream mining that includes many state-of-the-art stream generators, algorithms and evaluation methods.

Our previous and current work [19], developed under the Palco3.0 project (QREN AdI Palco3.0/2313), has focused on the development and evaluation of forgetting mechanisms (based on sliding windows and fading factors) in CF algorithms that deal with binary ratings. Currently we are working on the implementation of our developments on the Palco Principal website[1].

2 Objectives

In most recommender systems, usage data is continuously produced by user activity and can thus be looked at as a data stream. CF algorithms should be able to efficiently deal with these potentially unbounded streams of information. Also, as user preferences may change over time, these changes must be timely and effectively reflected in the model that is used to produce recommendations.

The aim of this doctoral project is to study and develop CF algorithms that are able to deal, in real time, with flows of information (data streams). These algorithms should provide accurate predictions as much as possible, and should be able to deal with high dimensional spaces, as required by current and future recommender system applications. This implies fast computation and low consumption of other computational resources, such as memory. They should also be sensitive to the dynamics of the incoming data and adapt their response to changes in users' preferences. These features will be evaluated on real data, and the algorithms will be compared with existing state-of-the-art competitors. If logistically possible, we will undertake an on-line evaluation of some of the proposed algorithms in a real web site. Our plan is to use www.palcoprincipal.pt, with whom we currently collaborate.

More specifically we intend to:

[1] http://www.palcoprincipal.com

- Develop scalable incremental CF algorithms using matrix factorization and/or clustering methods;
- Study and develop forgetting mechanisms that help focusing on more recent data;
- Study and develop drift detection mechanisms to perform automatic adaptation of forgetting parameters, especially to detect sudden changes in the distribution of the source data;
- Provide an off-line evaluation framework to enable reliable comparison between algorithms.

3 Detailed description

Our work plan is divided in four main tasks, explained below.

3.1 Task 1: Study the state-of-the-art in incremental CF, matrix factorization and stream mining

The aim of this task is to prepare an in depth report describing the state-of-the-art research in the fields of incremental CF, Matrix Factorization methods, Stream Clustering algorithms and Drift Detection mechanisms found in the literature. This report, along with a more detailed work plan, will be publicly presented and defended in the context of the enrolled doctoral program.

3.2 Task 2: Study and develop scalable incremental CF algorithms

One of the most effective ways to reduce the burden of model generation is to perform dimensionality reduction on usage data. Usage data is stored in an often very large and very sparse user vs item matrix. Using matrix factorization methods, it is possible to summarize data in a considerably smaller and denser matrix, with minimal loss of information. Similarities calculated using this smaller matrix are thus computed much faster than with the original data.

Because we intend to use incremental approaches, it is fundamental to select MF method(s) that allow incremental model updates. SVD is one of such methods, as explained in [17]. We also plan to study the potential of other MF methods that may have the potential to allow an incremental approach.

The potential of scalability improvements by using clustering models will also be studied during this task.

The task will be sub-divided in the following steps:

- Study existing and possible new incremental approaches to LSA/SVD, and other MF methods in the context of data stream processing;
- Study state-of-the-art stream clustering models and asses their potential to use in CF;
- Apply MF and/or clustering methods to incremental CF.

By the end of these steps, we expect to obtain a set of highly scalable, incremental algorithms.

3.3 Task 3: Study and implement automatic drift detection methodologies

In the field of data stream mining, many efforts have focused on the ability of algorithms to quickly adapt to new circumstances. In many today's applications, it is common that generated data streams suffer drifts in the underlying concepts, which requires models to adapt accordingly to maintain predictive accuracy.

In this task we intend to develop CF algorithms that are able to adaptively react to changes in user preferences, which are common, especially in long time frames. While some users tend to maintain their preferences, other may frequently change habits. Also, it is frequent that users maintain *certain* preferences while changing *other* preferences. For instance, a music fan may always like Bob Dylan, while at the same time have other more volatile tastes, such as getting tired of Oasis and start listening to Blur.

Our approach is to implement mechanisms that are able to detect these changes. This detection will then be used to dynamically and automatically trigger changes in the forgetting parameters that determine what gets "forgotten" by the model and at which rate.

The following steps are planned:

- Study and characterize drift phenomena in temporal data;
- Study drift detection mechanisms and methods;
- Implement and test drift detection in CF algorithms;
- Define and implement strategies to adaptively adjust forgetting parameters;
- Harmonize stream mining logistics with previous enhancements (MF, Clustering);

At the end of this task, the core of our development work will be completed. The obtained algorithms are expected to solve relevant issues of scalability and predictive ability.

3.4 Task 4: Evaluation

The main goal of this task is to build an off-line evaluation framework to perform an thorough evaluation of our work, in particular regarding contributions made during tasks 2 and 3. For this, detailed empirical experiments will be conducted. It is important to have an exact and accurate conclusion about every contribution made, that can also provide us with directions for future work.

The evaluation will consist of a set of methodologies, tools and scripts to:

- Automate tasks in evaluation experiments;
- Validate algorithms and datasets and detect implementation problems;
- Perform thorough evaluation of obtained algorithms.

This task can be further divided in the following steps:

- Select off-line evaluation protocol(s);
- Select and/or define metrics for accuracy measurement;
- Select datasets;
- Define an complete evaluation methodology;
- Choose and extend tools, such as MOA [2], and develop new ones to integrate and automate evaluation.

It is expected, in the course of the project duration, that the scientific community contributes with novel approaches to the same problems we face with this work. Many advances, using diverse approaches, have continuously been contributed in recent literature related to recommender systems. In this task, we also plan to make a comparative study between our work and other state-of-the-art developments.

3.5 Work plan

- Year 1: Task 1 (12 months)
- Year 2: Task 2 (10 months) + Task 3 (2 months)
- Year 3: Task 3 (12 months)
- Year 4: Task 4 (8 months) + Result analysis and thesis finishing (4 months)

Additionally, we intend to publish our work in relevant publications, according to the following plan:

- Year 1: 1 Conference or Workshop paper (e.g. ACM SAC, RecSys, Web Intelligence)
- Year 2: 2 Conference or Workshop papers (e.g. ACM SAC, RecSys, Web Intelligence)
- Year 3: 2 Conference or Workshop papers (e.g. ACM SAC, RecSys, Web Intelligence)
- Year 4: 1 Journal article (e.g. World Wide Web, Intelligent Data Analysis)

References

[1] Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. On demand classification of data streams. In Won Kim, Ron Kohavi, Johannes Gehrke, and William DuMouchel, editors, *KDD*, pages 503–508. ACM, 2004.

[2] Albert Bifet, Geoff Holmes, Richard Kirkby, and Bernhard Pfahringer. Moa: Massive online analysis. *Journal of Machine Learning Research*, 11:1601–1604, 2010.

[3] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[4] Yi Ding and Xue Li. Time weight collaborative filtering. In Otthein Herzog, Hans-Jörg Schek, Norbert Fuhr, Abdur Chowdhury, and Wilfried Teiken, editors, *CIKM*, pages 485–492. ACM, 2005.

[5] Pedro Domingos and Geoff Hulten. Catching up with the data: Research issues in mining data streams. In *DMKD '01: Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2001.

[6] João Gama, Raquel Sebastião, and Pedro Pereira Rodrigues. Issues in evaluation of stream learning algorithms. In John F. Elder IV, Françoise Fogelman-Soulié, Peter A. Flach, and Mohammed Javeed Zaki, editors, *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*, pages 329–338. ACM, 2009.

[7] Thomas George and Srujana Merugu. A scalable collaborative filtering framework based on co-clustering. In *ICDM 2005: Proceedings of the 5th IEEE International Conference on Data Mining, 27-30 November 2005, Houston, Texas, USA*, pages 625–628. IEEE Computer Society, 2005.

[8] Ralf Klinkenberg. Learning drifting concepts: Example selection vs. example weighting. *Intell. Data Anal.*, 8(3):281–300, 2004.

[9] Ralf Klinkenberg and Stefan Rüping. Concept drift and the importance of example. In *Text Mining*, pages 55–78. 2003.

[10] Yehuda Koren. Collaborative filtering with temporal dynamics. In John F. Elder IV, Françoise Fogelman-Soulié, Peter A. Flach, and Mohammed Javeed Zaki, editors, *KDD*, pages 447–456. ACM, 2009.

[11] Yehuda Koren, Robert M. Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

[12] I. Koychev. Gradual forgetting for adaptation to concept drift. In *Proceedings of ECAI 2000 Workshop Current Issues in Spatio-Temporal Reasoning*, pages 101–106, 2000.

[13] Tao Li and Sarabjot S. Anand. Hirel: An incremental clustering algorithm for relational datasets. In *ICDM*, pages 887–892. IEEE Computer Society, 2008.

[14] Catarina Miranda and Alípio Mário Jorge. Incremental collaborative filtering for binary ratings. In *Web Intelligence*, pages 389–392. IEEE, 2008.

[15] Olfa Nasraoui, Jeff Cerwinske, Carlos Rojas, and Fabio A. González. Performance of recommendation systems in dynamic streaming environments. In *Proceedings of the Seventh SIAM International Conference on Data Mining, April 26-28, 2007, Minneapolis, Minnesota, USA*. SIAM, 2007.

[16] Manos Papagelis, Ioannis Rousidis, Dimitris Plexousakis, and Elias Theoharopoulos. Incremental collaborative filtering for highly-scalable recommendation algorithms. In Mohand-Said Hacid, Neil V. Murray, Zbigniew W. Ras, and Shusaku Tsumoto, editors, *Foundations of Intelligent Systems, 15th International Symposium, ISMIS 2005, Saratoga Springs, NY, USA, May 25-28, 2005, Proceedings*, volume 3488 of *Lecture Notes in Computer Science*, pages 553–561. Springer, 2005.

[17] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Incremental SVD-based algorithms for highly scalable recommender systems. In *Fifth International Conference on Computer and Information Technology*, pages 27–28, 2002.

[18] Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. Investigation of various matrix factorization methods for large recommender systems. In *ICDM Workshops*, pages 553–562. IEEE Computer Society, 2008.

[19] João Vinagre and Alípio Mário Jorge. Forgetting mechanisms for incremental collaborative filtering. In *WTI 2010: Proceedings of the III Workshop on Web and Text Intelligence, October 23-28*, São Carlos, SP, Brazil, 2010.

[20] Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.

Phd Project Plan
João Vinagre

	Year 1 – 2012												Year 2 – 2013												Year 3 – 2014												Year 4 – 2015											
	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4	5	6	7	8	9	10	11	12
Task 1																																															
Task 2																																															
Task 3																																															
Task 4																																															
Result verification and thesis																																															

Submission plan												S1						S2						S3							S4						S5												S6