MyMediaLite: A Free Recommender System Library

Zeno Gantner
Machine Learning Lab
University of Hildesheim
gantner@ismll.de

Steffen Rendle
Social Network Analysis
University of Konstanz
steffen.rendle@uni-konstanz.de

Christoph Freudenthaler
Machine Learning Lab
University of Hildesheim
freudenthaler@ismll.de

Lars Schmidt-Thieme
Machine Learning Lab
University of Hildesheim
schmidt-thieme@ismll.de

ABSTRACT

MyMediaLite is a fast and scalable, multi-purpose library of recommender system algorithms, aimed both at recommender system researchers and practitioners. It addresses two common scenarios in collaborative filtering: *rating prediction* (e.g. on a scale of 1 to 5 stars) and *item prediction from positive-only implicit feedback* (e.g. from clicks or purchase actions). The library offers state-of-the-art algorithms for those two tasks. Programs that expose most of the library's functionality, plus a GUI demo, are included in the package. Efficient data structures and a common API are used by the implemented algorithms, and may be used to implement further algorithms. The API also contains methods for real-time updates and loading/storing of already trained recommender models.

MyMediaLite is free/open source software, distributed under the terms of the GNU General Public License (GPL). Its methods have been used in four different industrial field trials of the MyMedia project, including one trial involving over 50,000 households.

Categories and Subject Descriptors

H3.3 [**Information Search and Retrieval**]: Information Filtering—*Collaborative Filtering*

General Terms

Algorithms, Experimentation

Keywords

open source, e-commerce, personalization

1. INTRODUCTION

Free/open source implementations of recommender system algorithms are desirable for three reasons: (i) They relieve researchers from implementing existing methods for their experiments, either for comparing them against newly designed methods, or as recommendation methods in other kinds of studies, e.g. in user interface research, (ii) they can play a crucial role in the practical adoption of newly developed techniques, either by providing software that can be directly adapted and deployed, or by at least giving example implementations, (iii) and finally they can be used for (self-) teaching future recommender systems researchers and practitioners. Additionally, well-designed software frameworks can make the implementation and evaluation of new algorithms much easier.

MyMediaLite[1] aims to be such a framework. It targets both academic and industrial users, who may use the existing algorithms in the library, or use the framework for the implementation and evaluation of new algorithms.

2. FEATURES

MyMediaLite addresses two common scenarios in collaborative filtering: *rating prediction* (e.g. on a scale of 1 to 5 stars) and *item prediction from positive-only implicit feedback* (e.g. from clicks or purchase actions). It offers state-of-the-art algorithms for those two tasks, plus online-updates (where feasible), serialization of computed models, and routines for evaluation.

MyMediaLite is implemented in C#, and runs on the .NET platform. With the free .NET implementation Mono, it can be used on all relevant operating systems. Using the library is not limited to C#, though: it can be easily called from other languages e.g. Ruby and Python; code examples are included with the software.

2.1 Recommendation Tasks

Rating Prediction.

Ratings are a popular kind of explicit feedback. Users assess how much they like a given item (e.g. a movie or a news article) on a predefined scale, e.g. 1 to 5, where 5 could mean that the user likes the item very much, whereas 1 means the user strongly dislikes the item. *Rating prediction* algorithms estimate unknown ratings from a given set of known ratings and possibly additional data like user or item

[1]Here we describe MyMediaLite 1.04, released in August 2011.

attributes. The predicted ratings can then indicate to users how much they will like an item, or the system can suggest items with high predicted ratings.

MyMediaLite contains different variants of k-nearest neighbor (kNN) models [15], simple baseline methods (averages, Slope-One [14]), and modern matrix factorization methods [21] for the task of rating prediction.

Item Prediction from Positive-Only Implicit Feedback.

Rating prediction has been popularized in the research community by systems (and corresponding datasets) like MovieLens [8] and Jester [7], and later by the Netflix Prize [13]. Nevertheless, most real-world recommender systems (e.g. in e-commerce) do not rely on ratings, because users are hard to persuade to give explicit feedback, and other kinds of feedback (user actions like selecting/buying an item, page views/clicks, etc.) are often recorded by the system anyway. Buying a product or watching a video is also a positive expression of preferences. Note that not buying or watching an item from a large collection does not necessarily mean a user dislikes the item. *Item prediction from positive-only implicit feedback* is the task of determining the items that a user will perform a certain action on from such past events (and possibly additional data).[2]

The library contains kNN models for this task, as well as simple baselines (random/most popular item) and advanced matrix factorization methods like WR-MF [10] and BPR-MF [20].

Command-Line Tools.

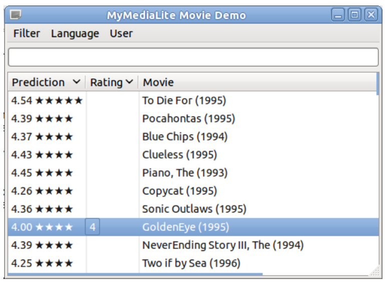
For each of the two recommendation tasks, MyMediaLite comes with a command-line program that allows users to train and evaluate all available recommenders on data provided in text files, without having to write a single line of code. Newly developed recommenders are automatically detected, and do not have to be manually added to the programs. Most of the other library features described in this section are exposed to the user by the command-line programs; if not, this is explicitly mentioned.

2.2 Data Sources

Besides collaborative data, that is the ratings in the case of rating prediction and the implicit user feedback in the case of item prediction, recommenders in MyMediaLite also may access other kinds of data: User (like geographic location, age, profession) or item attributes (categories, keywords, etc.), and relations between users (e.g. the social network) or items (taxonomies, TV series), respectively. Algorithms in the library that make use of this data are for example attribute-based kNN methods [3], a linear model optimized for BPR [5], and SocialMF, a matrix factorization model that takes the social network of the users into account [11].

MyMediaLite contains routines to read such data from SQL databases (not supported by the command-line programs) and from simple text files.

[2]For the sake of brevity, we will call this task *item prediction* in the following, even though there are of course other item predictions tasks besides item prediction from positive-only feedback.



Prediction	Rating	Movie
4.54 ★★★★★		To Die For (1995)
4.39 ★★★★		Pocahontas (1995)
4.37 ★★★★		Blue Chips (1994)
4.43 ★★★★		Clueless (1995)
4.45 ★★★★		Piano, The (1993)
4.26 ★★★★		Copycat (1995)
4.36 ★★★★		Sonic Outlaws (1995)
4.00 ★★★★	4	GoldenEye (1995)
4.39 ★★★★		NeverEnding Story III, The (1994)
4.25 ★★★★		Two if by Sea (1996)

Figure 1: The MyMediaLite movie demo program.

2.3 Additional Features

Evaluation.

MyMediaLite contains routines for computing evaluation measures [9] like root mean square error (RMSE) and mean average error (MAE) for the rating prediction task; for item prediction, it supports area under the ROC-curve (AUC), precision-at-N (prec@N), mean average precision (MAP), and normalized discounted cumulative gain (NDCG).

For rating prediction, internal k-fold cross-validation is supported, and there is also preliminary support for hyperparameter selection using grid search and Nelder-Mead [19].

Incremental Updates.

Academic experiments on recommender system algorithms are usually conducted off-line, by training a prediction model and then evaluating it. Yet real-world recommender systems constantly get new user feedback that should be immediately incorporated into the prediction model. MyMediaLite offers an API for immediate updates to already trained prediction models. Besides being supported in a special online evaluation mode by the two command-line tools, the use of incremental updates is demonstrated in the GUI demo which asks the user for movie ratings and then immediately displays personalized movie recommendations (see Fig. 1).

Serialization.

Another feature that is required in practice is storing and loading trained recommenders, which allows e.g. to train recommenders on a different machine than the one that produces the recommendations. All recommenders in MyMediaLite support this feature.

2.4 Developer Support

Documentation.

The library is accompanied by broad documentation: Besides the complete API documentation, there are example programs in Python, Ruby, and C#, and how-tos on typical tasks like embedding recommenders into a program, implementing new recommenders, or using the command-line tools.

Best Practices.

In the development of the library, we employ best practices for (not only) free/open source projects, e.g. keeping the code in a public (and distributed) version control repository, having regular releases (roughly one per month) and

Dataset	$k = 5$	$k = 120$
MovieLens-1M (CV)	61 MB	61 MB
Netflix (single split)	1297 MB	1734 MB

Table 2: Memory usage for rating prediction.

a collection of unit tests, and performing static analysis[3] of the compiled code.

3. RELATED WORK

Some researchers (e.g. [22]) have published their algorithms as free software. In this discussion, we limit ourselves to libraries that offer a wider choice of different recommendation methods.

Apache Mahout is a collection of mostly distributed implementations of machine learning and data mining algorithms. One section of the library is dedicated to collaborative filtering algorithms; the majority of algorithms (taken from the predecessor *Taste*) is not distributed; an item-based kNN model and Slope-One are available as a distributed implementation.

EasyRec is a recommender system web service that can be integrated into websites, however it does not contain any advanced personalized algorithms; it is more a framework for connecting a recommender service with an application. *RecLab* is a framework for performing live evaluations in online shopping systems; it contains an API to be implemented by the shop system, and another one for providing recommendations.

We collected information on some other recommender system algorithm libraries' features and compare them to those of MyMediaLite in Table 1. We concentrated on free software/open source[4], and additionally included SUGGEST [12], which is not free software, but was a fairly early publicly available software.

4. EXPERIMENTS

We have performed several rating prediction experiments to showcase MyMediaLite's runtime performance and scalability. For this, we ran the *BiasedMatrixFactorization* recommender on the *Netflix* and *MovieLens-1M* datasets. For *Netflix*, we used the *probe* dataset for validation, on the *MovieLens* dataset we performed 5-fold cross-validation. We measured the time needed for one pass over the training, the time needed for predicting the validation set, and the memory usage reported by the program. Measuring the time needed for a pass over the training data should give a general idea of the performance of the employed data structures, not just the particular recommender. The evaluation scripts with all relevant parameter settings are available from `http://ismll.de/mymedialite/examples/recsys2011.html`.

MyMediaLite is capable of making between 1,400,000 ($k = 120$) and 7,000,000 ($k = 5$) predictions per second.

We have not performed a systematic comparison with other libraries yet. [2] reports on experiments involving several recommender system frameworks.

[3]We use *Gendarme* to find problems in MyMediaLite.

[4]See `http://www.gnu.org/philosophy/free-sw.html` and `http://www.opensource.org/docs/osd`.

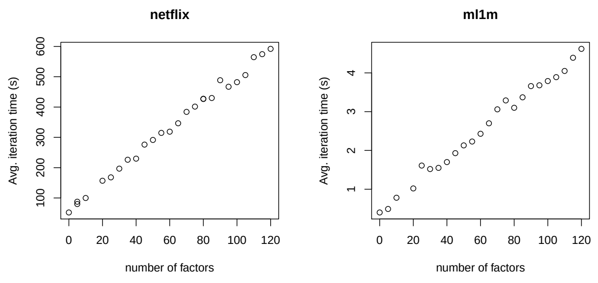


Figure 2: Runtime of *BiasedMatrixFactorization*.

5. MYMEDIALITE USAGE

MyMediaLite is based on parts of the framework [16] that has been used in four different industrial field trials of the MyMedia project, including one involving 50,000 households [17]. Application areas in the field trials were IPTV, web-based video and audio, and online shopping. Additionally, the MyMedia framework was used to perform user-centric experiments [4].

Towards the end of the project, we stripped off the framework's more heavyweight components, and released it as free software in September 2010. Since then, it has been downloaded more than 2,300 times (excluding search engine spider downloads), received numerous improvements, and has been used in several research activities, e.g. studies on student performance prediction [18], cold-start (new item) problems [5], and as a baseline for context-aware recommendation (winner of two tracks of the CAMRa 2010 challenge [6]).

6. CONCLUSIONS

We have presented MyMediaLite, a versatile library of recommender system algorithms for rating and item prediction from positive-only feedback. We believe MyMediaLite is currently one of the most complete free/open source recommender system frameworks in terms of recommendation tasks, implemented methods, efficiency, features, flexibility, and documentation (see section 3).

We will continue MyMediaLite's development in several directions. Porting the library to Java or C++ is worth considering, given the popularity of these programming languages. A Java port of a part of the library is already available for download. Another useful extension would be a web service interface for the easy integration of MyMediaLite e.g. into online shop software; we will consider existing APIs for this feature. We also plan to add additional recommendation tasks and types of input, e.g. item prediction from other kinds of implicit feedback like viewing times or click counts, or tag recommendation.

Finally, we would like to invite both researchers and practitioners in the field of recommender systems to use and extend MyMediaLite, and to give us feedback for possible improvements.

7. ACKNOWLEDGMENTS

The development of MyMediaLite was co-funded by the European Commission FP7 project MyMedia under the grant agreement no. 215006. We thank Chris Newell of BBC R+D

Library	Duine	MultiLens	LensKit	Mahout	GraphLab	SUGGEST	MyMediaLite
version	4.0.0-RC1	–	0.04	0.4	v1_134	1.0	1.02
date	2009-02-17	2004-10-22	2011-07-29	2010-10-31	2011-07-29	2000-11-08	2011-08-03
actively developed	–	–	✓	✓	✓	–	✓
license	LGPL 3	GPL	LGPL 2	Apache 2.0	Apache 2.0	*non-free*	GPL 3
language	Java	Java	Java	Java	C++	C	C#
scalable	–	✓	✓	✓	✓	✓	✓
matrix factorization	–	–	✓	✓	✓	–	✓
rating prediction	✓	✓	✓	✓	✓	–	✓
implicit feedback	–	–	–	–	–	✓	✓
online updates	✓	–	–	✓	–	–	✓

Table 1: Comparison of free/open source recommender system frameworks.

Software	**URL**
Duine	`http://duineframework.org`
GraphLab	`http://graphlab.org`
LensKit	`http://lenskit.grouplens.org`
Mahout	`http://mahout.apache.org`
MultiLens	`http://knuth.luther.edu/~bmiller/dynahome.php?page=multilens`
MyMediaLite	`http://ismll.de/mymedialite`
RecLab	`http://code.richrelevance.com`
SUGGEST	`http://glaros.dtc.umn.edu/gkhome/suggest/overview`
Gendarme	`http://www.mono-project.com/Gendarme`
Mono	`http://www.mono-project.com`

Table 3: Websites of software packages mentioned in the paper.

for giving valuable feedback about MyMediaLite, and for porting parts of the library to Java. We also thank all contributors to MyMediaLite. They are listed in the README file of the package.

8. REFERENCES

[1] X. Amatriain, M. Torrens, P. Resnick, and M. Zanker, editors. *Proceedings of the 2010 ACM Conference on Recommender Systems, (RecSys 2010)*. ACM, 2010.

[2] T. Angermann. Empirische Analyse von Open-Source-Empfehlungssystemen. Master's thesis, University of Hildesheim, 2010.

[3] D. Billsus, M. J. Pazzani, and J. Chen. A learning agent for wireless news access. In *Intelligent User Interfaces*, pages 33–36, 2000.

[4] D. Bollen, B. P. Knijnenburg, M. C. Willemsen, and M. Graus. Understanding choice overload in recommender systems. In Amatriain et al. [1].

[5] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recommendations. In G. I. Webb, B. Liu, C. Zhang, D. Gunopulos, and X. Wu, editors, *ICDM 2010*. IEEE Computer Society, 2010.

[6] Z. Gantner, S. Rendle, and L. Schmidt-Thieme. Factorization models for context-/time-aware movie recommendations. In *Challenge on Context-aware Movie Recommendation (CAMRa2010)*. ACM, 2010.

[7] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm, 2000.

[8] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *SIGIR 1999*, New York, NY, USA, 1999. ACM.

[9] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22:5–53, January 2004.

[10] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM 2008*, 2008.

[11] M. Jamali and M. Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In Amatriain et al. [1].

[12] G. Karypis. Evaluation of item-based top-n recommendation algorithms. In *CIKM*, 2001.

[13] Y. Koren, R. M. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8), 2009.

[14] D. Lemire and A. Maclachlan. Slope one predictors for online rating-based collaborative filtering. In *Proceedings of SIAM Data Mining (SDM'05)*, 2005.

[15] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7, 2003.

[16] P. Marrow, R. Hanbidge, S. Rendle, C. Wartena, and C. Freudenthaler. MyMedia: Producing an extensible framework for recommendation. In *NEM Summit*, 2009.

[17] P. Marrow, T. Stevens, I. Kegel, J. Meenowa, and C. McCahill. Future IPTV services field trial report. Technical report, MyMedia project, 2010.

[18] T.-N. Nguyen, L. Drumond, T. Horvath, A. Nanopoulos, and L. Schmidt-Thieme. Matrix and tensor factorization for predicting student performance. In *Proceedings of the 3rd International Conference on Computer Supported Education (CSEDU 2011)*, 2011.

[19] M. Piotte and M. Chabbert. The Pragmatic Theory solution to the Netflix Grand Prize, 2009.

[20] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *UAI 2009*, 2009.

[21] S. Rendle and L. Schmidt-Thieme. Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In *RecSys '08: Proceedings of the 2008 ACM Conference on Recommender Systems*. ACM, 2008.

[22] M. Weimer, A. Karatzoglou, and A. Smola. Improving maximum margin matrix factorization. *Machine Learning*, 72(3):263–276, 2008.