Collaborative filtering with recency-based negative feedback

João Vinagre
FCUP - Universidade do Porto
LIAAD - INESC TEC
Porto, Portugal
jnsilva@inesctec.pt

Alípio Mário Jorge
FCUP - Universidade do Porto
LIAAD - INESC TEC
Porto, Portugal
amjorge@fc.up.pt

João Gama
FEP - Universidade do Porto
LIAAD - INESC TEC
Porto, Portugal
jgama@fep.up.pt

ABSTRACT

Many online communities and services continuously generate data that can be used by recommender systems. When explicit ratings are not available, rating prediction algorithms are not directly applicable. Instead, data consists of positive-only user-item interactions, and the task is therefore not to predict ratings, but rather to predict good items to recommend – *item prediction*. One particular challenge of positive-only data is how to interpret absent user-item interactions. These can either be seen as negative or as unknown preferences. In this paper, we propose a recency-based scheme to perform negative preference imputation in an incremental matrix factorization algorithm designed for streaming data. Our results show that this approach substantially improves the accuracy of the baseline method, outperforming both classic and state-of-the-art algorithms.

Categories and Subject Descriptors

H.3.3 [**Information Search-Retrieval**]: Information Filtering

Keywords

Recommender Systems, Data streams, Incremental

1. INTRODUCTION

The task of recommendation algorithms for item prediction consists of distinguishing between good and bad recommendations, when only positive examples are available for training – i.e. negative feedback is absent. This problem is also known as One-Class Collaborative Filtering (OCCF), given its similarity to One-class Classification [3]. One way to tackle problems related with positive-only feedback is to artificially introduce negative examples. In this paper, we propose a strategy to select negative feedback based on the recency of item occurrence in a stream of user-item positive interactions. This strategy is implemented in an incremental matrix factorization algorithm designed for streaming data.

SAC'15 April 13-17, 2015, Salamanca, Spain

ACM 978-1-4503-3196-8/15/04...$15.00.
http://dx.doi.org/10.1145/2695664.2695998

2. INCREMENTAL MATRIX FACTORIZATION

The most popular method for recommender algorithms is matrix factorization. Given a conceptual users *vs.* items ratings matrix R, the main task is to obtain two factor matrices A and B covering a latent feature space common to both users and items. Matrix A spans the user space and matrix B spans the item space, such that every cell in R can be obtained by the dot product $R_{ui} = A_u B_i^T$. Training is performed by minimizing the L_2-regularized squared error associated with known values in R and the corresponding predicted ratings:

$$\min_{A.,B.} \sum_{(u,i)\in D} (R_{ui} - A_u.B_i^T)^2 + \lambda(||A_u||^2 + ||B_i||^2) \qquad (1)$$

In the above equation, D is the set of user-item pairs for which ratings are known and λ is a parameter that controls the amount of regularization. The regularization term $\lambda(||A_u||^2 + ||B_i||^2)$ is used to avoid overfitting.

The most popular method to solve this optimization problem is Stochastic Gradient Descent (SGD). While this process is typically done in batch – performing multiple iterations through learning dataset –, in [9] we use an incremental approach, continuously adjusting a factorised model over a stream of positive-only user feedback. For each observed user-item pair (u, i), the error associated with the prediction is minimized for that pair. The rating value 1 is assumed for observed examples, and the prediction error is measured as $e_{ui} = 1 - A_u.B_i^T$. Recommendations for any user u are obtained by sorting all candidate items i by a function $f = |1 - A_u.B_i^T|$ that basically calculates the absolute proximity to value 1. This simple algorithm yields competitive accuracy, when compared with other incremental alternatives.

3. NEGATIVE FEEDBACK

Given that in our problem setting we need to learn a predictive model by analysing positive-only user-item preferences, the problem is how to distinguish between the two possible interpretations of *absent* user-item interactions in the feedback data. If an interaction between a user and an item does not occur, this can either be interpreted as a candidate preference – the user does not know the item – or as negative preference – the user does not like the item. However, it is not trivial to make a clear distinction between negative and candidate items when only positive examples are available. It is possible to find in the literature some important contributions, using sampling and weighting [5, 2], and a graph-based approach [6]. These contributions, like ours, provide techniques to select negative feedback from missing user-item pairs. The fundamental difference between our approach and the aforementioned is that while these are essentially batch procedures, we use a recency-based scheme in an incremental algorithm designed for streaming data.

3.1 Recency-based negative feedback

One problem of our baseline approach described in Sec. 2 is that the absence of negative examples leads to a model that converges globally to the positive class, eventually causing accuracy degradation. This is a consequence of using an algorithm that is originally designed for ordinal ratings, that approaches the problem as a regression task, not accounting for the absence of negative examples. As discussed in Sec. 3, one way to solve this problem is to artificially introduce negative feedback. The problem is then how to select the negative feedback from the (u, i) pairs that do not occur in the data. Our strategy is to select a set of negative examples for each observed (positive) example, based on the recency of occurrence of items in the stream. For every observed (u, i) in the data stream, we introduce a set $\{(u, j_1), \ldots, (u, j_l)\}$ of negative feedback consisting of the active – currently observed – user u and the l items j that occurred the farthest back in the data stream. We implement this scheme in Alg. 1 by maintaining a FIFO (First-In-First-Out) queue containing all items seen so far. Every time an item occurs, it is moved to – or inserted at, if new – the head of the queue. The l items in the tail of the queue are selected for negative feedback at each step. To avoid penalizing infrequent items repeatedly, when an item is used as negative feedback it is also moved to the head of the queue.

Algorithm 1: RA-ISGD: Recency-Adjusted ISGD

Data: $D = \{(u, i)\}$, a finite set or a data stream
input : $feat$, λ, η, l
output: A, B

$Q \leftarrow \mathtt{initqueue}()$
for $(u, i) \in D$ **do**
  if $u \notin \mathtt{Rows}(A)$ **then**
    $A_u \leftarrow \mathtt{Vector}(\mathsf{size}: feat)$
    $A_u \sim \mathcal{N}(0, 0.1)$
  if $i \notin \mathtt{Rows}(B^T)$ **then**
    $B_i^T \leftarrow \mathtt{Vector}(\mathsf{size}: feat)$
    $B_i^T \sim \mathcal{N}(0, 0.1)$
  for $k \leftarrow 1$ **to** $\mathtt{min}(l, \#Q)$ **do**
    $j \leftarrow \mathtt{dequeue}(Q)$
    $err_{uj} \leftarrow 0 - A_u.B_j^T$
    $A_u \leftarrow A_u + \eta(err_{uj}B_j - \lambda A_u)$
    $\mathtt{enqueue}(Q, j)$
  $err_{ui} \leftarrow 1 - A_u.B_i^T$
  $A_u \leftarrow A_u + \eta(err_{ui}B_i - \lambda A_u)$
  $B_i \leftarrow B_i + \eta(err_{ui}A_u - \lambda B_i)$
  if $i \in Q$ **then**
    $\mathtt{remove}(Q, i)$
  $\mathtt{enqueue}(Q, i)$

In Alg. 1, the hyper-parameters $feat$, η and λ are respectively the number of features, learn rate and regularization factor. The actual recency-based adjustment is done by following the gradient of the error – with respect to the negative class 0 – associated with a set of items in the tail of the FIFO queue and the active user. The length of this set is given by the user defined parameter l. The queue related functions `initqueue()`, `enqueue()`, `dequeue()` and `remove()` respectively perform queue initialization, head insertion, tail removal, and index-based removal.

The update operations corresponding to negative feedback only change the user factor matrix, leaving the item factor matrix unmodified. We have verified empirically that updating item features with negative feedback is actually harmful to the model's predictive ability. This is possibly explained by the intrinsic stability of items relative to users. However, this discussion is beyond the scope of this article.

Dataset	Events	Users	Items	Sparsity
Music-listen	335.731	4.768	15.323	99,90%
Lastfm-600k	493.063	164	65.013	99,11%
Music-playlist	111.942	10.392	26.117	99,96%
MovieLens-1M	226.310	6.014	3.232	98,84%

Table 1: Dataset description

4. EXPERIMENTAL SETUP AND RESULTS

To evaluate RA-ISGD, we the datasets described in Table 1. All four datasets consist of a chronologically ordered set of user-item pairs in the form (u, i). Music-listen and Lastfm-600k consist of music listening events and Music-playlist is a record of music track additions to personal playlists. MovieLens-1M is a well known dataset[1] consisting of movie ratings. Since we intend to retain only positive feedback, movie ratings below the maximum rating 5 are excluded. Lastfm-600k is a subset of the Last.fm[2] dataset collected by Celma [3]. Both Music-listen and Music-playlist are extracted from the Palco Principal[4] website.

We use prequential evaluation [8] and measure accuracy with recall at cutoff 10. For each observed event (u, i), representing a positive interaction between user u and item i, we generate 10 item recommendations for user u. We then score the recommendation list with 1 if the true observed item i is within the recommendations, and 0 otherwise. Then the pair (u, i) is used to update the model and we move on to the next data point.

To avoid cold-start issues – which are not the subject of our research – we perform an initial batch training using the first 20% data points in each dataset. Parameters are independently tuned for each algorithm on the initial 20% of data of each dataset. We compare the accuracy of Alg. 1 against three other incremental algorithms for CF: BPRMF [7] – a state-of-the.art ranking optimization algorithm –, UKNN [4] – a classic incremental neighborhood-based algorithm – and ISGD [9] – our baseline method. The optimal values for the parameter l in RA-ISGD are $l = 1$ for Music-playlist and Lastfm-600k, $l = 3$ for Music-listen and $l = 10$ for Movielens-1M.

We depict the accuracy of the learning process by plotting

[1]http://www.grouplens.org, 2003
[2]http://last.fm
[3]http://ocelma.net/MusicRecommendationDataset, 2014
[4]http://www.palcoprincipal.com

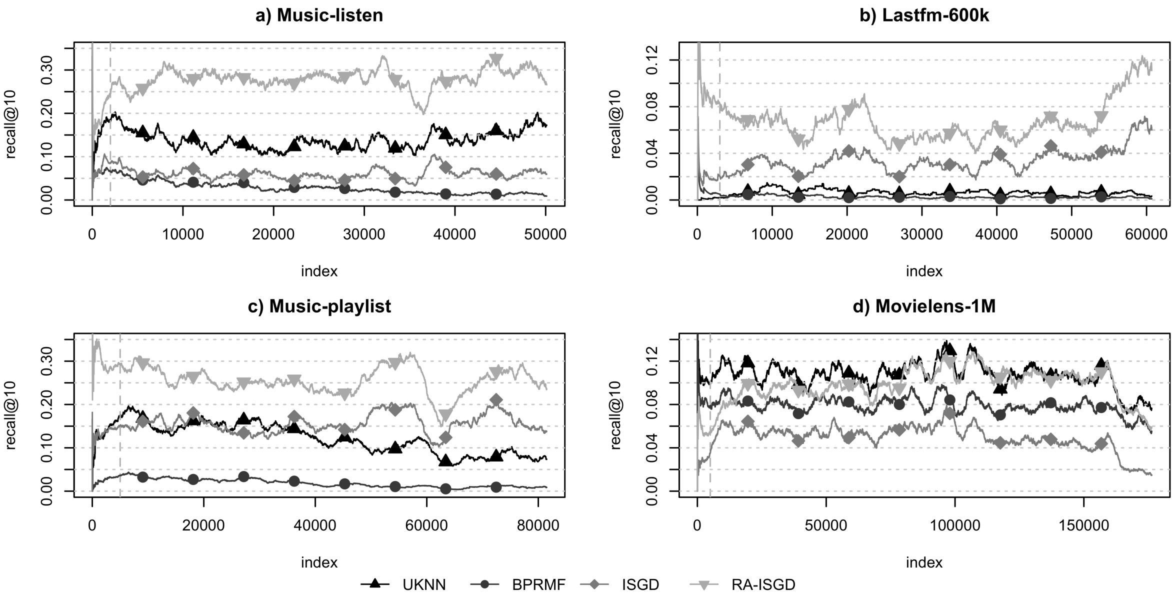

Figure 1: Evolution of recall@10. Lines are plotted using a moving average of the recall@10. The window size n of the moving average is a) $n = 2000$, b) $n = 3000$, c) $n = 5000$ and d) $n = 5000$. The first n points delimited by the vertical dashed line and are plotted using the accumulated average.

the evolution of recall over time in Fig. 1. One first observation is the clear improvement of RA-ISGD over its baseline ISGD. RA-ISGD outperforms all other algorithms in all datasets except Movielens-1M. However, closely looking at Fig. 1 d), although RA-ISGD starts worse than UKNN, it converges to an accuracy level quite similar to UKNN.

5. CONCLUSIONS

This work addresses one important challenge of recommender systems using positive-only data, which is the absence of negative examples. We propose an incremental recency-based scheme to automatically select negative examples and use them to update the model. Our experiments suggest that this method substantially improves the accuracy of our baseline method that does not use negative feedback imputation. Moreover, it considerably outperforms other state-of-the-art incremental methods in 3 datasets while still being competitive in one dataset.

6. ACKNOWLEDGEMENTS

This work is partially funded by National Funds through FCT - Fundação para a Ciência e Tecnologia (proj. NORTE-07-0124-FEDER-000059) and by the European Commission through proj. MAESTRA (Grant no. ICT-2013-612944). The first author is supported by FCT with grant SFRH / BD / 77573 / 2011. We also thank Ubbin Labs, Lda. for kindly providing data from Palco Principal.

7. REFERENCES

[1] *Proceedings of the 8th IEEE Intl. Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*. IEEE Computer Society, 2008.

[2] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM* [1], pages 263–272.

[3] S. S. Khan and M. G. Madden. One-class classification: taxonomy of study and review of techniques. *Knowledge Eng. Review*, 29(3):345–374, 2014.

[4] C. Miranda and A. M. Jorge. Incremental collaborative filtering for binary ratings. In *WI 2008, IEEE / WIC / ACM International Conference on Web Intelligence, 9-12 December 2008, Sydney, NSW, Australia*, pages 389–392. IEEE Computer Society, 2008.

[5] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. M. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *ICDM* [1], pages 502–511.

[6] U. Paquet and N. Koenigstein. One-class collaborative filtering with random graphs. In *WWW '13, 22nd International World Wide Web Conference, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 999–1008. ACM, 2013.

[7] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI 2009, 25th Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, pages 452–461. AUAI Press, 2009.

[8] J. Vinagre, A. M. Jorge, and J. Gama. Evaluation of recommender systems in streaming environments. In *Workshop on Recommender Systems Evaluation: Dimensions and Design, 8th ACM Conf. on Recommender Systems (RecSys 2014), Foster City, CA, USA, October 10*, 2014.

[9] J. Vinagre, A. M. Jorge, and J. Gama. Fast incremental matrix factorization for recommendation with positive-only feedback. In *UMAP 2014, 22nd Conference on User Modeling, Adaptation and Personalization, June 7-11, Aalborg, Denmark*, volume 8538 of *LNCS*, pages 459–470. Springer, 2014.