Sequential Collaborative Ranking Using (No-)Click Implicit Feedback

Frédéric Guillou, Romaric Gaudel, and Philippe Preux

Inria Lille - Nord Europe, Univ. Lille, CRIStAL (UMR CNRS)
Villeneuve d'Ascq, France
{frederic.guillou,romaric.gaudel,philippe.preux}@inria.fr

Abstract. We study Recommender Systems (RS) in the context where it suggests a list of items to users. Several crucial issues are raised in such a setting: first, which ranking of the items should be provided to put best items for the user at the top of the list? Second, how to account for the feedback given by the user after he clicked and rated an item? Third, since new feedback arrive into the system at any moment, how to incorporate such information to improve future recommendations? In this paper, we take these three aspects into consideration and present an approach handling click/no-click feedback information. Several experiments on real-world datasets show that our approach outperforms several state of the art algorithms.

1 Introduction

Recommender Systems (RS) seek to suggest to users items they might like or need. The recommendation builds upon the feedback given by users at previous recommendations. In this paper, we focus on RS aiming to recommend a list of items. Standard approaches in that setting discuss which loss is the most adapted and how to optimize that loss [12,1,10]. In their attempts to do so, they omit one effect deriving from recommending a list of items: we could expect more feedback than a rating on a single item. For example, we could have access to the list of items which have been seen by the user, the list of items on which he clicked, etc. In short, we are confronted to a mix of explicit and implicit feedback.

The feedback gathered depends on the model of interaction for users. The first contribution of the paper is to propose two approaches to handle the feedback involved by the interaction model discussed in [3]. These two approaches lead to relevant recommendation lists for three real-life datasets.

The second contribution of our paper relates to the experimental setting. The data used by a RS depend on previous recommendations. This means two different RS should recommend based on two different sets of feedback. As a consequence, the evaluation of the RS traditionally done in a fixed, batch setting does not make sense. In this paper, we propose a proper experimental setting which mimics the interaction between the RS and the users.

The paper is organized as follows. We first go through the related work in Sec. 2. Then, in Sec. 3, we specify the setting of the recommendation process

and we present our two approaches. Sec. 4 provides an experimental study on real datasets. Finally, we conclude and draw some future lines of work in Sec. 5.

2 Related Work

In this section, we briefly review state of the art in RS. Among approaches recommending a list of items, we omit the ones only handling implicit data due to space limitations.

Approaches recommending a list of items are looking at a the best loss function. These approaches replace the squared loss on ratings by a loss measuring the ranking ability of the model learned. Examples of such models include [12] and [1] which both optimize a smooth approximation of the *Normalized Discounted Cumulative Gain* (NDCG), or [10], [8], and [6] which respectively optimize smooth approximations of the *Expected Reciprocal Rank* (ERR), of a pairwise learning to rank loss, and of a structured output loss. [11] also optimizes a pairwise learning to rank loss, but it adds a first step of feature construction.

These approaches only handle either explicit or implicit feedback. However, when recommending a list of items, a mix of both kinds of feedback is often to be expected: the rating of selected items (explicit), the list of items which have not been clicked despite having been shown to the user (implicit), the list of items which have been selected but not purchased, etc.

SVD++ [5] and Co-Rating [7] mix explicit and implicit data to build their RS. On the one hand, SVD++ integrates implicit feedback in the representation of a user and limits itself to a restrictive kind of implicit feedback: a list of clicked items. On the other hand, Co-Rating considers implicit feedback as complementary values to fit. As such, Co-Rating manages any kind of implicit data. Still, it assumes that explicit and implicit feedback equal themselves as soon as they are scaled to $[0, 1]$. SVD++ and Co-Rating both learn by optimizing a squared loss.

In the remainder of the paper, we develop an approach which properly handles any kind of feedback, and therefore leads to better experimental results.

3 Ranking Recommender System Using Click Feedback

We now propose two approaches to handle both explicit and implicit feedback. Before that presentation, we have to express and define the model of interaction we will use in what follows. This model governs the feedback which are gathered, and is used to build our second approach (Sec 3.2).

We consider the recommendation setting described in [3]. Let us consider N users and M items and the unknown matrix $\mathbf{R}^*$ of size $N \times M$ such that $r^*_{i,j}$ is the rating of user i with regard to item j. We assume the ratings range from 0 to R. From $\mathbf{R}^*$, we derive a relevance probability $p(i, j)$ which represents how much a user i is eager to select an item j:

$$p(i,j) = \frac{2^{r^*_{i,j}} - 1}{2^R}. \tag{1}$$

Then we set up the interaction between the RS, the users and the items. At each time-step t, a user i_t requests ℓ items. The RS provides the ranked list $j_1^{(t)}, \ldots, j_\ell^{(t)}$, one item at a time, starting with the top-ranked item. While observing the s-ith item, the user has a chance to pick it with probability $p(i_t, j_s^{(t)})$. Once an item is picked (denoted j_t), the user stops observing the list, and reveals the rating $r^*_{i_t,j_t}$. Thus, note that the user does not observe following items.

This setting implies two types of feedback are received at every recommendation list displayed: the first one is the list of skipped items and the clicked item (aka. implicit feedback), and the second one is the rating of the clicked item, if there is one (aka. explicit feedback). Note that the rank (in the recommendation list) of the clicked item brings only information about items ranked above in the list: we know these items were skipped. On the contrary, no information is gathered about items placed below the clicked one, as the user did not observe them according to our setting. The following parts describe two different approaches attempting to use these types of feedback to improve recommendations.

In the following, we denote $\mathbf{A}$ and $\mathbf{S}$ the matrices of size $N \times M$ which respectively stores for each user i and item j the number of clicks received by item j when presented to i, and the number of times j was skipped when shown to i on a recommendation list. We also note $\mathcal{S}$ the set of known entries of $\mathbf{R}^*$.

3.1 Feature Engineering

The first method denoted SVD+- assumes we can embed the implicit feedback into features of the model. Following the data representation used in Factorization Machine (FM) [9], we associate to any user-item couple (i, j) a representation $\phi(i, j)$ and we look at a function f *s.t.* $f(\phi(i, j)) = r^*(i, j)$. We take

$$\phi(i,j) = (\underbrace{0,\ldots,1,0,\ldots}_{N},\underbrace{0,\ldots,1,0,\ldots}_{M},\underbrace{\mathbf{C}_{i1},\ldots,\mathbf{C}_{is},\ldots,\mathbf{C}_{iM}}_{M}), \tag{2}$$

where the first section indexes i, the second section indexes j, and

$$\mathbf{C}_{is} = \frac{\mathbf{A}_{is} - \mathbf{S}_{is}}{\sum_{s'=1}^{M} \mathbf{A}_{is'} + \mathbf{S}_{is'}}. \tag{3}$$

$\mathbf{C}$ gives a value to every item based on how many times it was clicked or skipped out of all interactions for a user. This value ranges from -1 to 1.

From that feature model, the Factorization Machine learns the function

$$\hat{f}(\phi(i,j)) = w_0 + w_i + w_j + v_u.v_j^T + \left(\sum_{s=1}^{M} C_{is}v_s\right)\cdot v_j^T + v_u \cdot \left(\sum_{s=1}^{M} C_{is}v_s\right)^T + \sum_{s=1}^{M} C_{is}w_s + \left(\sum_{s=1}^{M} C_{is}v_s\right)\cdot\left(\sum_{s=1}^{M} C_{is}v_s\right)^T,$$

where w_0, w_i, w_j and $(w_s)_{1\leqslant s\leqslant M}$ are real values, and v_0, v_i, v_j and $(v_s)_{1\leqslant s\leqslant M}$ are feature vectors of size k. These parameters are chosen to optimize a trade-off

between (i) the average square loss with respect to known values in $\mathbf{R}^*$ and (ii) the L_2 norm of the parameters.

3.2 Dual Matrix Factorization

From another perspective, we design a second approach called DualMF, which considers both types of feedback as values to fit. More specifically, we look at a low rank approximation $\widehat{\mathbf{R}} = \mathbf{U}.\mathbf{V}^T$ of $\mathbf{R}^*$, where $\mathbf{U}$ and $\mathbf{V}$ are matrices of respective sizes $N \times k$ and $M \times k$. Obviously, we want $\widehat{\mathbf{R}}$ to fit known values in $\mathbf{R}^*$ (aka. explicit feedback).

But we also make use of implicit feedback. Based on the model of the probability of click $p(i,j)$, we can build a biased estimator $\hat{r}_{ij}^{imp}$ of r_{ij}^* for any user-item couple (i,j) *s.t.* item j has been clicked or skipped at least one time by user i:

$$\hat{r}^{imp}(ij) = \log_2 \left(1 + 2^R \frac{\mathbf{A}_{ij} + 0.5}{\mathbf{A}_{ij} + \mathbf{S}_{ij} + 1}\right). \tag{4}$$

The 0.5 added at the numerator and the 1 added at the denominator act similarly to a prior and help the model to not penalize too much items for which only little information has yet been collected.

Overall, the approach DualMF looks for a matrix $\widehat{\mathbf{R}}$ fitting both known values in $\mathbf{R}^*$, and $\hat{r}_{ij}^{imp}$ values. It solves the minimization problem:

$$\min_{\widehat{\mathbf{R}}=\mathbf{U}.\mathbf{V}^T} \mu \sum_{(i,j)\in\mathcal{S}} (\hat{r}_{ij} - r_{ij}^*)^2 + \sum_{(i,j):A_{ij}+Sij\neq 0} (\hat{r}_{ij} - \hat{r}_{ij}^{imp})^2 + \lambda(\|\mathbf{U}\|^2 + \|\mathbf{V}\|^2), \tag{5}$$

where μ and λ are non-negative real values. μ controls the impact of explicit data compared to implicit one, and λ weights the regularization term.

4 Experimental investigation

We empirically evaluate the algorithms in a sequential setting on real-world datasets. For each dataset, we start with an empty matrix $\mathbf{R}$ to simulate an extreme cold-start scenario where no information is available at all. Then, a list of items is recommended for each user according to the following procedure:

1. we select a user i_t uniformly at random among possible users (the ones to which no recommendation list was displayed yet),
2. the algorithm chooses a list of 5 items to recommend,
3. the user observes the list, clicks or not on an item j_t according to the setting described in Sec. 3. The value of r_{i_t,j_t} is revealed if there was a click. The user is then discarded from possible users.

When all users have been shown a recommendation list once, we reintegrate all of them in the list of possible users, and loop on this procedure again, while keeping all feedback gathered until now. These steps are done up to 50 recommendations

shown for every user. As the ground truth is unknown for every item, we restrict the possible choices for a user at each time-step to the items with a known rating in the dataset. Note that it is allowed to include an item in the list of recommendations for a user even if it has already been rated in the past by him. Metrics used for the evaluation are abandonment, ERR@5 and NDCG@5. Let us consider a user i, to which the list of items $j_1, \ldots, j_\ell$ has been recommended. The abandonment metric represents the probability $\prod_{r=1}^{\ell}(1-p(i,j_r))$ for a user not to be satisfied by any item in the list (no click), while the ERR and NDCG depict how adequate is the list of items suggested by the RS regarding the user's tastes. The ERR is based on the click probability defined by Eq. (1):

$$ERR@\ell(i,(j_1,\ldots,j_\ell)) = \sum_{r=1}^{\ell} \frac{1}{r} \prod_{s=1}^{r-1}(1-p(i,j_s)p(i,j_r). \tag{6}$$

The formula used for the Discounted Cumulative Gain at ℓ [2] is:

$$DCG@\ell(i,(j_1,\ldots,j_\ell)) = \sum_{r=1}^{\ell} \frac{2^{r^*_{ij_r}}-1}{\log_2(r)}. \tag{7}$$

The NDCG is the DCG divided by the best possible DCG. As we have no access to the ground-truth matrix $\mathbf{R}^*$, the NDCG is computed *w.r.t.* the known values.

We consider three real-world datasets for our experiments: Movielens1M, Yahoo! Music ratings for User Selected and Randomly Selected songs, and a subset of Yahoo! Music user ratings of musical artists. To build this subset, we first select the 10,000 most rated artists, and then the 50,000 users who rated the highest number of artists. We also normalize the ratings from the range 0-100 to 1-5 and put the ratings with 255 (meaning "do not recommend this ever again") to 0.5. Characteristics of these datasets are reported in Table 1.

Movielens1M is a standard dataset to compare RS approaches, but it contains mostly high ratings. On the other hand, the Yahoo! datasets contain a lot more low ratings, which allows for more realistic experiments (since items to recommend are chosen only among the ones for which we have ratings).

Our two approaches are compared to the following baselines:

- Oracle: this strategy knows the ground truth matrix and makes recommendation accordingly, with the list of best possible items for each user.
- Random: at each iteration, a random list of items is recommended to the user.
- Popular: we assume we know the most popular items based on the ground-truth matrix, and at each iteration, the 5 most popular items (restricted to the items rated by the user on the dataset) are recommended.

Table 1. Dataset characteristics.

	Movielens1M	Yahoo! songs	Yahoo! artists
Number of users	6,040	15,400	50,000
Number of items	3,706	1,000	10,000
Number of ratings	1,000,209	365,703	32,997,016

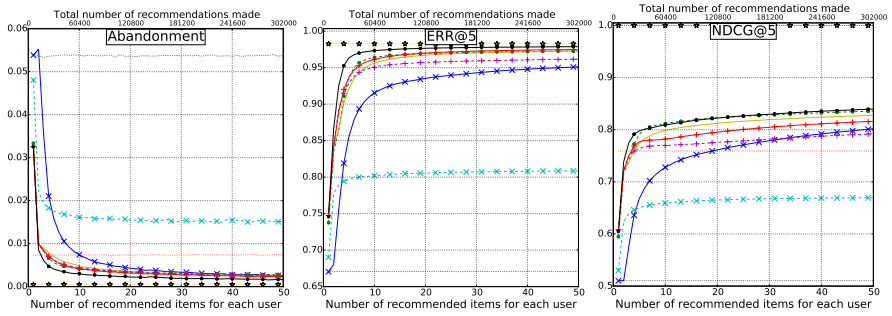
- xCliMF [10]: this model is built by optimizing the ERR on explicit ratings. It does not use implicit features but targets an appropriate ranking of items.
- CoFiRank [12]: CoFiRank optimizes a ranking measure using explicit features. In our experiments, we use the version optimizing the ordinal regression.
- ALS: we use a Factorization Machine implementation of Alternating Least Square method, taking into account only the explicit ratings given by users.
- SVD++ [5]: this approach also consists in a FM with data arranged to make the model mimic SVD++ (while adding pairwise interactions between features), as described in [9]. It also incorporates both explicit data and implicit data, but only implicit data about the item clicked.
- Co-Rating [7]: this approach tries to unify explicit and implicit feedback. To do so, it first normalizes both explicit and implicit scores between 0 and 1 and combines them when solving the minimization problem, using ALS. The main difference with DualMF is that it does not assume any model behind the observed clicks. In order to compare fairly with DualMF, we also use a FM to learn the model, and choose the implicit score to be $\hat{r}^{imp}(ij) = \frac{\mathbf{A}_{i,j}+0.5}{\mathbf{A}_{i,j}+\mathbf{S}_{i,j}+1}$.

Since most state of the art algorithms are not designed to handle the sequential aspect which implies frequent updates, we update the model of each approach each time 50% of the users have been recommended a list of items (overall, each algorithm will have 100 updates spread along the 50 recommendations). Results of all algorithms on the datasets are shown on Fig. 1. We use existing implementation called fastFM for models using Factorization Machines (ALS, SVD++, SVD+-, Co-Rating, DualMF). For Co-Rating and DualMF, we give different weight to explicit and implicit feedback when training the model, and best results are obtained by giving twice more weight to explicit feedback for DualMF and five times more weight to explicit feedback for Co-Rating.

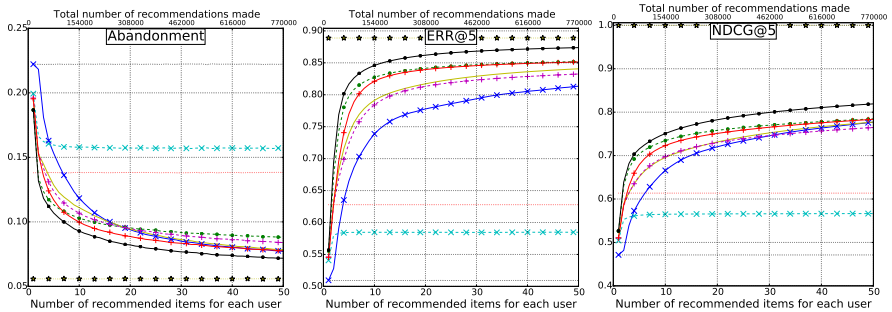
From the results of experiments, we can draw two main conclusions: firstly, the approaches learning from both explicit and implicit feedback (DualMF, Co-Rating and SVD+-) perform significantly better than all other approaches on all datasets. Our approach DualMF in particular reach the best performance on Movielens1M and Yahoo! songs, and is on par with Co-Rating on Yahoo! artists except on the NDCG@5 metric. Note that the implicit score given during the learning phase by DualMF comes directly from the click model we defined. In practice, it would not be possible to access such knowledge and it would have to be inferred. Results of these three methods emphasizes the importance of using any feedback given by the user. Secondly, approaches targeting specifically a good ranking by optimizing ranking loss (xCliMF and CoFiRank) surprisingly performs the worst, even compared to those using only explicit feedback like ALS. Aiming at the solving the learning to rank aspect does not seem to be a priority to reach a good performance.

5 Conclusion and future work

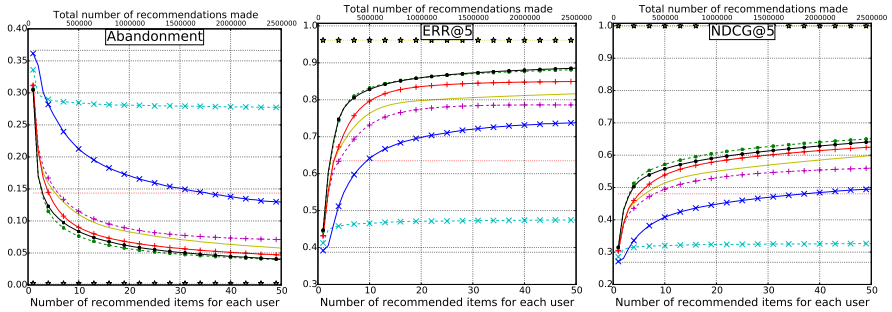
We study Recommender Systems from a novel point of view, where at every step a list of recommendation is provided to the user, and the system receives both



(a) Movielens1M



(b) Yahoo! songs



(c) Yahoo! artists

Fig. 1. Evaluation of algorithms on three datasets and three metrics (from the left column to the right one: Abandonment, ERR@5, NDCG@5). Plotted values correspond to the average score obtained while recommending a list of items to each user. For all algorithms learning matrices of features to represent users and items, we fix the number of columns of these matrices to 15. All algorithms using FM use a L2 penalty weight of $\lambda = 5.0$ for both pairwise coefficients and linear coefficients, and do one more step of learning at each update. CoFiRank and xCliMF do from 20 to 50 steps of learning at each update depending on the dataset, but relearn from scratch due to implementation.

explicit and implicit feedback. This model of interaction impacts the learning algorithm but also the experimental setting. We provide a case study for a specific interaction model for which we propose two approaches, tackling the problem from two distinct perspectives. We evaluate various state of the art methods on several metrics, and results on experiments display how considering both implicit and explicit feedback can significantly improve the performance.

Several aspects could be targeted as future work. First, would it be possible to infer the click model directly as new feedback is received? Second, a sequential setting also raises concerns about finding good balance between gathering information about the user and providing good recommendations, a.k.a. the exploration-exploitation dilemma [4], which we did not consider in this paper. We leave the attempt to incorporate a solution about this dilemma in the proposed approaches as future work.

References

1. Balakrishnan, S., Chopra, S.: Collaborative ranking. In: Proc. of the fifth ACM int. conf. on Web Search and Data Mining (WSDM'12). pp. 143–152. ACM (2012)
2. Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., Hullender, G.: Learning to rank using gradient descent. In: Proc. of the 22nd int. conf. on Machine Learning (ICML'05). pp. 89–96. ACM (2005)
3. Chapelle, O., Metlzer, D., Zhang, Y., Grinspan, P.: Expected reciprocal rank for graded relevance. In: Proc. of the 18th ACM conf. on Information and knowledge management (CIKM'09). pp. 621–630. ACM (2009)
4. Kawale, J., Bui, H., Kveton, B., Thanh, L.T., Chawla, S.: Efficient thompson sampling for online matrix-factorization recommendation. In: NIPS'15 (2015)
5. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 426–434. ACM (2008)
6. Lee, J., Bengio, S., Kim, S., Lebanon, G., Singer, Y.: Local collaborative ranking. In: Proc. of the 23rd int. conf. on World Wide Web (WWW'14). pp. 85–96 (2014)
7. Liu, N.N., Xiang, E.W., Zhao, M., Yang, Q.: Unifying explicit and implicit feedback for collaborative filtering. In: Proceedings of the 19th ACM international conference on Information and knowledge management. pp. 1445–1448. ACM (2010)
8. Liu, X., Aberer, K.: Towards a dynamic top-n recommendation framework. In: Proc. of the 8th conf. on Recommender Systems (RecSys'14). pp. 217–224 (2014)
9. Rendle, S.: Factorization machines. In: Data Mining (ICDM), 2010 IEEE 10th International Conference on. pp. 995–1000. IEEE (2010)
10. Shi, Y., Karatzoglou, A., Baltrunas, L., Larson, M., Hanjalic, A.: xclimf: optimizing expected reciprocal rank for data with multiple levels of relevance. In: Proceedings of the 7th ACM conference on Recommender systems. pp. 431–434. ACM (2013)
11. Volkovs, M., Zemel, R.S.: Collaborative ranking with 17 parameters. In: Advances in Neural Information Processing Systems. pp. 2294–2302 (2012)
12. Weimer, M., Karatzoglou, A., Le, Q.V., Smola, A.J.: Cofi rank - maximum margin matrix factorization for collaborative ranking. In: Advances in Neural Information Processing Systems 20. pp. 1593–1600 (2008)