A Preliminary Study on a Recommender System for the Million Songs Dataset Challenge

Fabio Aiolli[1]

Abstract. In this paper the preliminary study we are conducting on the Million Songs Dataset (MSD) challenge is described. The task of the competition is to suggest a set of songs to a user given half of its listening history and complete listening history of other 1 million people. We focus on memory-based collaborative filtering approaches since they are able to deal with large datasets in an efficient and effective way. In particular, we investigated on *i)* defining suitable similarity functions, *ii)* studying the effect of the locality of the collaborative scoring function, and *iii)* aggregating multiple ranking strategies to define the overall recommendation. Using these techniques we are in the top positions according to the current standing of the competition leaderboard (at the moment of this writing the challenge has about 150 registered teams).

1 Introduction

The Million Song Dataset Challenge [5] is a large scale, music recommendation challenge, where the task is to predict which songs a user will listen to, provided the listening history of the user. The challenge is based on the Million Song Dataset (MSD), a freely-available collection of meta data for one million of contemporary songs (e.g. song titles, artists, year of publication, audio features, and much more) [2]. About one hundred and fifty teams are currently participating to the challenge. The subset of data actually used in the challenge is the so called Taste Profile Subset that consists of more than 48 million triplets *(user,song,count)* gathered from user listening histories. Data consists of about 1.2 million users and covers more than 380,000 songs in MSD. The user-item matrix is very sparse as the fraction of non-zero entries (the density) is only 0.01%.

Collaborative Filtering (CF) is a technology which uses the items by user matrix to discover other users with similar tastes as the active user for which we want to make the prediction. The intuition is that if other users, similar to the active user, already purchased a certain item, then it is likely that the active user will like that item as well. A similar (dual) consideration can be made by changing the point of view. If we know that a set of items are often purchased together (they are similar in some sense), then, if the active user has bought one of them, probably he/she will be interested to the other as well. The first view is the one that is prevalent in recent CF literature. In this paper, we show that the second view turned out more useful when used for the MSD competition.

In Section 2 collaborative filtering is described and proposed as a first approach to solve the problem of MSD. In the same section three different views of the memory-based CF task are proposed that motivated the different algorithms of the section. In Section 3 empirical results of the proposed techniques are presented and discussed.

2 A CF approach to the MSD task

Collaborative filtering techniques use a database in the form of a user-item matrix R of preferences. In a typical CF scenario a set $\mathcal{U}$ of n users and a set $\mathcal{I}$ of m items exist and the entries of $R = \{r_{ui}\} \in \mathbb{R}^{n\times m}$ represent how much user u likes item i. In this paper, we assume $r_{ui} \in \{0,1\}$ as this is the setting of the MSD challenge. Entries r_{ui} represent the fact that user u have listened to (or would like to listen to) the song i. In the following we refer to items or songs interchangeably. The MSD challenge task is more properly described as a *top-τ recommendation* task. Specifically, we want to identify a list of τ ($\tau = 500$ in the challenge) items $I_u \subseteq \mathcal{I}$ that *active user* u will like the most. Clearly, this set must be disjoint with the set of items already rated (purchased, or listened to) by the active user.

2.1 Memory-based CF

In memory-based CF algorithms the entire user-item matrix is used to generate a prediction. Generally, given a new user for which we want to obtain the prediction, the set of items to suggest are computed looking at similar users. This strategy is typically referred to as *user-based recommendation*. Alternatively, in the *item-based recommendation* strategy, one computes the most similar items for the items that have been already purchased by the active user, and then aggregates those items to form the final recommendation. There are many different proposal on how to aggregate the information provided by similar users/items (see [7] for a good survey). We focus on the simple weighted sum strategy. A deeper analysis of this simple strategy allows us to highlight an interesting duality that exists between user-based and item-based recommendation algorithms.

In the *user-based* type of recommendation, the scoring function, on the basis of which the recommendation is made, is computed by

$$h^U_{ui} = \sum_{v\in\mathcal{U}} f(w_{uv}) r_{vi} = \sum_{v\in\mathcal{U}(i)} f(w_{uv}),$$

that is, the score obtained on an item for a target user is proportional to the similarities between the target user u and other users v that have purchased the item i ($v \in \mathcal{U}(i)$). This score will be higher for items which are often rated by similar users.

On the other hand, within a *item-based* type of recommendation [3, 6], the target item i is associated with a score

$$h^S_{ui} = \sum_{j\in\mathcal{I}} f(w_{ij}) r_{uj} = \sum_{j\in\mathcal{I}(u)} f(w_{ij}),$$

and hence, the score is proportional to the similarities between item i and other items already purchased by the user u ($j \in \mathcal{I}(u)$).

[1] University of Padova, Italy, email: aiolli@math.unipd.it

The function $f(w)$ can be assumed monotonic not decreasing and its role is to emphasize/deemphasize similarity contributions in such a way to adjust the *locality* of the scoring function, that is how many of the nearest users/items really matter in the computation. As we will see, a correct setting of this function turned out to be very useful with the challenge data.

Interestingly, in both cases, we can decompose the user and item contributions in a linear way, that is, we can write $h^U_{ui} = \mathbf{w}_u^\top \mathbf{r}_i$, $\mathbf{w}_u \in \mathbb{R}^n$, and $h^S_{ui} = \mathbf{w}_i^\top \mathbf{r}_u$, $\mathbf{w}_i \in \mathbb{R}^m$. In other words, we are defining an embedding for users (in user based recommendation systems) and for items (in item based recommendation systems). In the specific case above, this corresponds to choose the particular vector $\mathbf{r}_i$ as the vector with n entries in $\{0,1\}$, where $\mathbf{r}_u^{(i)} = r_{ui}$. Similarly, for the representation of users in item-based scoring, we choose $\mathbf{r}_u$ as the vector with m entries in $\{0,1\}$, such that $\mathbf{r}_i^{(u)} = r_{ui}$. One of the main contributions of this paper is to explore how we can set the vectors $\mathbf{w}_i$ and $\mathbf{w}_u$ in a principled way. However, it is simple to imagine how we can try to learn the weight vectors from data. In fact, this recommendation task can also be seen as a multilabel classification problem where songs represent the labels and users represent the examples. We have performed preliminary experiments in this sense using the preference learning approach described in [1]. The results are promising but the problem in this case is the computational requirements of a *model-based* paradigm like this. For this reason we decided to postpone a further analysis of this setting to future works.

Finally, an alternative and useful way to see at the duality between user-based and item-based recommendation is the following that highlight a clear connection between this task and link prediction. Specifically, we can think of the user-item matrix $R = \{r_{ui}\}$ as a bipartite graph where the set of nodes is $\mathcal{N} = \mathcal{U} \cup \mathcal{I}$, and there exist only edges from $u \in \mathcal{U}$ to $i \in \mathcal{I}$ whenever $r_{ui} = 1$. Now, the user based recommendation strategy corresponds to the intuition that if a user tends to link to the same set of items as the active users, then, this gives us some evidence that can exist a link between the active user and the item i. Dually, in item-based recommendation, the intuition is that, if the active user links to one out of two items which tend to be linked by the same users, then, we can infer that the active user will probably link the other item as well.

2.2 User-based and Song-based similarity

A large part of CF literature in the last decade deals with the problem of defining a good similarity measure. A common opinion is that it cannot exist a single similarity measure that can fit all possible domains where collaborative filtering is used. In this section, we try to define a parametric family of user-based and item-based similarities that can fit different problems.

In the challenge, we have not relevance grades since the ratings are binary values. This is a first simplification we can exploit in the definition of similarity functions. The similarity function that is commonly used in this case, both for the user-based case and the item-based case, is the cosine similarity. In the case of binary grades the cosine similarity can be simplified as in the following. Let $\mathcal{I}(u)$ be the set of items rated by a generic user u, then the cosine similarity between two users u, v is defined by

$$w_{uv} = \frac{|\mathcal{I}(u) \cap \mathcal{I}(v)|}{|\mathcal{I}(u)|^{\frac{1}{2}} |\mathcal{I}(v)|^{\frac{1}{2}}}$$

and, similarly for items, by setting $\mathcal{U}(i)$ the set of users which have rated item i, we have:

$$w_{ij} = \frac{|\mathcal{U}(i) \cap \mathcal{U}(j)|}{|\mathcal{U}(i)|^{\frac{1}{2}} |\mathcal{U}(j)|^{\frac{1}{2}}}.$$

The cosine similarity has the nice property to be symmetric but, as we show in the experimental section, it might not be the better choice. In fact, especially for the item case, we are more interested in computing how likely it is that an item will be liked by a user when we *already* know that the same user likes another item. It is clear that this definition is not symmetric. As an alternative to the cosine similarity and, we think, a more well founded way of computing weights w_{ij} in this case, is by resorting to the conditional probability measure which can be estimated with the following formulas:

$$w_{uv} = P(u|v) = \frac{|\mathcal{I}(u) \cap \mathcal{I}(v)|}{|\mathcal{I}(v)|}$$

and

$$w_{ij} = P(i|j) = \frac{|\mathcal{U}(i) \cap \mathcal{U}(j)|}{|\mathcal{U}(j)|}$$

Previous works (see [4] for example) pointed out that the conditional probability measure of similarity, $P(i|j)$, has the limitation that items which are purchased frequently tend to have higher values not because of their co-occurrence frequency but instead because of their popularity. In our opinion, this might not be a limitation in a recommendation setting. Perhaps, this could be an undesired feature when we want to cluster items. In fact, this correlation measure has not to be thought of as a real similarity measure. As we will see, experimental results seem to confirm this hypothesis, at least in the item-item similarity case.

Now, we are able to propose a parametric generalization of the above similarity measures. This parameterization permits us ad-hoc optimizations of the similarity function for the domain of interest. For example, this can be done by validating on available data.

In particular, we propose to use the following combination of conditional probabilities:

$$w_{uv} = P(v|u)^\alpha P(u|v)^{1-\alpha} \quad w_{ij} = P(j|i)^\alpha P(i|j)^{1-\alpha} \qquad (1)$$

where $\alpha \in [0,1]$ is a parameter to tune. As above, we estimate the probabilities by resorting to the frequencies in the data and derive the following:

$$w_{uv} = \frac{|\mathcal{I}(u) \cap \mathcal{I}(v)|}{|\mathcal{I}(u)|^\alpha |\mathcal{I}(v)|^{1-\alpha}} \quad w_{ij} = \frac{|\mathcal{U}(i) \cap \mathcal{U}(j)|}{|\mathcal{U}(i)|^\alpha |\mathcal{U}(j)|^{1-\alpha}}. \qquad (2)$$

It is easy to note that the standard similarity based on the conditional probability $P(u|v)$ (resp. $P(i|j)$) is obtained setting $\alpha = 0$, the other inverted conditional $P(v|u)$ (resp. $P(j|i)$) is obtained setting $\alpha = 1$, and, finally, the cosine similarity case is obtained when $\alpha = \frac{1}{2}$. This analysis also suggests an interesting (and novel, to our knowledge) interpretation of the cosine similarity on the basis of conditionals.

2.3 Locality of the Scoring Function

In Section 2 we have seen that, in memory based CF, the final recommendation is computed by a scoring function which aggregates the scores obtained using individual users or items. So, it is important to determine how much each individual scoring component influences the overall scoring. This is the role of the function $f(w)$. In the following experiments we use the exponential family of functions, that is $f(w) = w^q$ where $q \in \mathbb{N}$. The effect of this exponentiation is the following. When q is high, smaller weights drop to zero while higher ones are (relatively) emphasized. At the other extreme, when $q = 0$, the aggregation is performed by simply adding up the ratings. We can note that, in the user-based type of scoring function, this corresponds to take the popularity of an item as its score, while, in the case of item-based type of scoring function, this would turn out in a constant for all items (the number of rating made by the active user).

2.4 Ranking Aggregation

There are many sources of information available regarding songs. For example, it could be useful to consider the additional meta-data which are also available and to construct alternative rankings based on that. It is always difficult to determine a single strategy which is able to correctly rank the songs. An alternative is to use multiple strategies, generate multiple rankings, and finally combine those rankings. Typically, these different strategies are individually *precision oriented*, meaning that each strategy is able to correctly recommend a few of the correct songs with high confidence but, it may be that, other songs which the user likes, cannot be suggested by that particular ranker. Hopefully, if the rankers are different, then the rankers can recommend different songs. If this is the case, a possible solution is to predict a final recommendation that contains all the songs for which the single strategies are more confident. A stochastic version of this aggregation strategy can be described in the following way. We assume we are provided with the list of songs not yet rated by the active user in order of confidence for all the available strategies. On each step, the recommender randomly choose one of the lists according to a probability distribution p_i over the predictors and recommends the best scored item of the list which has not yet been inserted in the current recommendation.

3 Experiments and Results

In the MSD challenge we have: *i)* the full listening history for about 1M users, *ii)* half of the listening history for 110K users (10K validation set, 100K test set), and we have to predict the missing half. Further, we also prepared a "home-made" validation subset (*HV*) of the original training data of about 900K users of training (*HVtr*, with full listening history). The remaining 100K user's histories has been split in two halves (*HVvi* the visible one, *HVhi* the hidden one).

The experiments presented in this section are based on this *HV* data and compare different similarities and different approaches. The baseline is represented by the simple popularity based method which recommends the most popular songs not yet listened to by the user. Besides the baseline, we report experiments on both the user-based and song-based scoring functions, and an example of the application of ranking aggregation.

3.1 Taste Profile Subset Stats

For completeness, in this section, we report some statistics about the original training data. In particular, the following table shows the minimum, maximum, and average, number of users per song and songs per user. The median value is also reported.

.	min	max	ave	median
users per song	1	110479	125.794	13
songs per user	10	4400	47.45681	27

We can see that the large majority of songs have only few users which listened to it (less than 13 users for half of the songs) and the large majority of users have listened to few songs (less than 27 for half of the users). These characteristics of the dataset make the top-τ recommendation task quite challenging.

3.2 Truncated Mean Average Precision

Conformingly to the challenge, we used the truncated mAP (mean average precision) as the evaluation metric. Let y denote a ranking over items, where $y(p) = i$ means that item i is ranked at position p. The mAP metric emphasizes the top recommendations. For any $k \leq \tau$, the *precision at k* (π_k) is defined as the proportion of correct recommendations within the top-k of the predicted ranking (assuming the ranking y does not contain the visible songs),

$$\pi_k(u, y) = \frac{1}{k} \sum_{p=1}^{k} r_{uy(p)}$$

For each user the (truncated) average precision is the average precision at each recall point:

$$AP(u, y) = \frac{1}{\tau_u} \sum_{p=1}^{\tau} \pi_k(u, y) r_{uy(p)}$$

where τ_u is the smaller between τ and the number of user u's positively associated songs.

The average of $AP(u, y_u)$'s over all users gives the mean average precision (mAP).

3.3 Results

The result obtained on the HV data with the baseline (recommendation by popularity) is presented in Table 1. With this strategy, each song i simply gets a score proportional to the number of users $|\mathcal{U}(i)|$ which listened to the song.

Baseline (Recommendation by Popularity)	0.02262

Table 1: mAP@500 obtained by the baseline method (song popularity) on HV data.

In Table 2, we report on experiments which show the effect of the locality parameter q for different strategies: item based and user based (both conditional probability and cosine versions). As we can see, beside the case IS with cosine similarity (Table 2b), a correct setting of the parameter q drammatically improves the effectiveness on HV data. We can clearly see that the best performance is reached with the conditional probability on an item based strategy (Table 2a).

In Figure 1, we present results obtained fixing the parameter q and varying the parameter α for both user-based and item-based recommendation strategies. We see that, in the item-based case, the results improve when setting a non-trivial α. The best result has been obtained for $\alpha = 0.15$.

Finally, in Table 3, two of the best performing rankers are combined, and their recommendation aggregated, as described in Section 2.4. In particular, in order to maximize the diversity of the two rankers, we aggregated an item-based ranker with a user-based ranker. We can see that the combined performance improves further on validation data.

IS ($\alpha = 0$)	mAP@500
q=1	0.12224
q=2	0.16581
q=3	**0.17144**
q=4	0.17004
q=5	0.16830

(a)

IS ($\alpha = \frac{1}{2}$)	mAP@500
q=1	**0.16439**
q=2	0.16214
q=3	0.15587
q=4	0.15021
q=5	0.14621

(b)

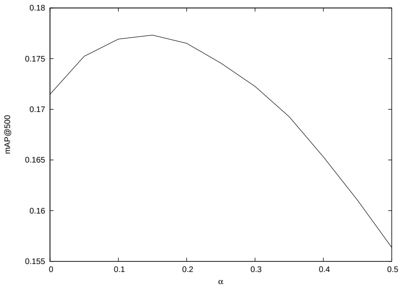
US ($\alpha = 0$)	mAP@500
q=1	0.08030
q=2	0.10747
q=3	0.12479
q=4	0.13298
q=5	**0.13400**
q=6	0.13187
q=7	0.12878

(c)

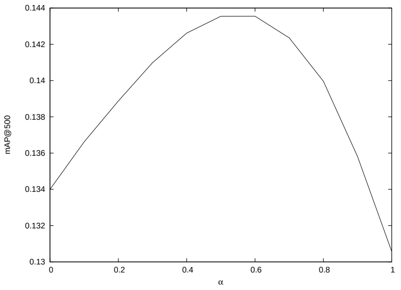
US ($\alpha = \frac{1}{2}$)	mAP@500
q=1	0.07679
q=2	0.10436
q=3	0.12532
q=4	0.13779
q=5	0.14355
q=6	**0.14487**
q=7	0.14352

(d)

Table 2: Results obtained by item-based (IS) and user-based (US) CF methods varying the locality parameter (exponent q) of the similarity function.



(a) IS with $0 \leq \alpha \leq 0.5$, $q = 3$, best-mAP@500: $0.177322 (\alpha = 0.15)$



(b) US with $0 \leq \alpha \leq 1$, $q = 5$, best-mAP@500: $0.143551 (\alpha = 0.6)$

Figure 1: Results obtained by item-based (IS) and user-based (US) CF methods varying the α parameter.

(IS, $\alpha = 0.15$, $q = 3$)	(US, $\alpha = 0.3$, $q = 5$)	mAP@500
0.0	1.0	0.14098
0.1	0.9	0.14813
0.2	0.8	0.15559
0.3	0.7	0.16248
0.4	0.6	0.16859
0.5	0.5	0.17362
0.6	0.4	0.17684
0.7	0.3	0.17870
0.8	0.2	**0.17896**
0.9	0.1	0.17813
1.0	0.0	0.17732

(a)

Table 3: Results obtained aggregating the rankings of two different strategies, item-based (IS, $\alpha = 0.15$, $q = 3$) and user-based (US, $\alpha = 0.3$, $q = 5$), with different combinations.

ACKNOWLEDGEMENTS

We would like to thank the referees for their comments, which helped improve this paper considerably

REFERENCES

[1] Fabio Aiolli and Alessandro Sperduti, ‘A preference optimization based unifying framework for supervised learning problems’, in *Preference Learning*, eds., Johannes Fürnkranz and Eyke Hüllermeier, 19–42, Springer-Verlag, (2010).

[2] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere, ‘The million song dataset’, in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, (2011).

[3] Mukund Deshpande and George Karypis, ‘Item-based top-*n* recommendation algorithms’, *ACM Trans. Inf. Syst.*, **22**(1), 143–177, (2004).

[4] George Karypis, ‘Evaluation of item-based top-n recommendation algorithms’, in *CIKM*, pp. 247–254, (2001).

[5] Brian McFee, Thierry Bertin-Mahieux, Daniel P.W. Ellis, and Gert R.G. Lanckriet, ‘The million song dataset challenge’, in *Proceedings of the 21st international conference companion on World Wide Web*, WWW ’12 Companion, pp. 909–916, New York, NY, USA, (2012). ACM.

[6] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John Riedl, ‘Item-based collaborative filtering recommendation algorithms’, in *WWW*, pp. 285–295, (2001).

[7] Xiaoyuan Su and Taghi M. Khoshgoftaar, ‘A survey of collaborative filtering techniques’, *Advances in Artificial Intelligence*, (January 2009).