# Action Prediction Models for Recommender Systems Based on Collaborative Filtering and Sequence Mining Hybridization

Tural Gurbanov
Free University of Bozen-Bolzano
Bozen-Bolzano, Italy
tgurbanov@unibz.it

Francesco Ricci
Free University of Bozen-Bolzano
Bozen-Bolzano, Italy
fricci@unibz.it

## ABSTRACT

Many recommender systems collect online users' activity and infer from it users' preferences. They record user actions of various types (e.g. clicks, views), and predict unknown, possibly future, interactions between users and items, mostly using Collaborative Filtering (CF) or Sequence Mining (SM) techniques. While both techniques have their advantages, in this paper, we show that improved prediction accuracy can be achieved by hybridizing them. The proposed hybrid model uses first an SM model to augment an existing actions' data set and then uses collaborative filtering in the final prediction step. The empirical evaluation, which was conducted on a large real-world dataset, showed that the proposed hybrid model outperforms both stand-alone SM and CF.

## CCS Concepts

•**Information systems** → **Recommender systems;**

## Keywords

Hybrid recommender systems, implicit feedback, collaborative filtering, sequence mining

## 1. INTRODUCTION

Recommender Systems (RSs) are software tools and techniques providing personalized suggestions for items that are predicted to be of interest to the user. Recommendations are adapted to the known user's preferences, which are either explicitly expressed, e.g., in the form of ratings for items, or are inferred by interpreting online user activity [21]. High-quality *explicit feedback* is considered to be the most informative type of source data [7]. However, in many real-world scenarios this type of feedback can be difficult to obtain or unavailable (e.g., news portals). Hence, more and more recommenders are built by leveraging abundant *implicit feedback* data, such as the log of item/page views or item purchases, which only indirectly signal users' preferences or opinions [14]. Crucial features and problems, which are raised by the usage of implicit feedback, such as no negative feedback, inherent noise, indication of confidence, need of appropriate measure, are described in [7].

Various approaches have been developed to build recommender systems employing only implicit feedback data. The largest number of these approaches are based on collaborative filtering (CF) techniques [15, 19]. CF identifies recommendations for a user by comparing her profile, which is a model of the user's preferences and interests, with the profiles of other users and then proposes the items which like-minded users like. Other approaches, especially when the user actions have temporal patterns, utilize sequence mining (SM) techniques [8, 13]. These techniques analyze the sequence of a user's previously consumed items and try to predict the most appropriate next item (e.g., music playlist generation).

In some cases, the system collects various types of, possibly repeated, user actions. For instance, before an item is bought by a user, it might be searched or browsed by the same user several times. In this way, for each item, one can observe a sequence of different actions performed by a single user. But, most of the CF models that have been developed so far ignore information about the sequence of actions performed by a user on an item, and the user-item interaction is described with a single numeric score (e.g., the user clicked on the item a certain number of times) or an indication (the user clicked the item at least once). Such representation, while simplifying the prediction model design, ignores valuable information and, for instance, does not take into account temporal delays between actions or the order of actions. On the other hand, SM models traditionally do not combine information about like-minded users, and just exploit commonalities between sequences of actions.

In this paper, we conjecture that a combination of SM and CF can be beneficial and may help to overcome the above-mentioned drawbacks of the two basic models. For this reason, we introduce a hybrid recommender system [3] integrating SM and CF models. The proposed hybrid system predicts whether a user will perform an action of a target type on an item, which is in our application the reply to a job posting. The system leverages observations of a range of actions of different types, which are performed by the user on the target item and by other users on other items. The SM component considers the ordering of the actions and the



*SAC 2017, April 03 - 07, 2017, Marrakech, Morocco*



ISBN 978-1-4503-4486-9/17/04…$15.00

DOI: http://dx.doi.org/10.1145/3019612.3019759

time delays between them, while the CF component employs collaboration between multiple items and users.

Moreover, standard SM models predict a target action on an item given a sequence of observed actions on other items. In contrast, the SM model we have developed here predicts the probability that a user will perform an action of a target type on an item, given the sequence of the user's previous actions on the item. The estimated probabilities are then used as input for a CF model that predicts the observation of the target action on items. In fact, in this last step, the user-action data is extended with the probabilities of observing the target action on the items where the user has not performed the target action yet. The experimental evaluation, which was conducted on a dataset from the ACM RecSys Challenge 2016[1], shows that the proposed hybrid approach outperforms pure SM and CF models and state-of-the-art implicit feedback models.

The rest of the paper is organized as follows. In Section 2, we review previous works related to the proposed hybrid system. Then, to ease the illustration of the prediction models, we describe the dataset and the data representation in Section 3 and Section 4 respectively. In Section 5, we introduce the system prediction models. The evaluation protocol and its results are provided in Section 6 and Section 7. Finally, in Section 8, we conclude the paper and indicate some future works.

## 2. RELATED WORKS

CF predicts unobserved user-item interactions, such as ratings or clicks, by exploiting users-to-users or items-to-items similarities. Two concrete implementations of CF are nearest neighborhood (NN) and matrix factorization (MF) models [6, 9]. In NN the interaction predictions are generated by aggregating observed user-item interactions of users (items) similar to the target user (item). In MF each user $u$ and item $i$ is mapped to an $f$-dimensional factors vector $x_u \in \mathbb{R}^f$ and $y_i \in \mathbb{R}^f$. The unknown user-item interactions are predicted by computing the dot product of their corresponding user and item vectors, $x_u$ and $y_i$.

In implicit feedbacks data sets, the most popular NN and MF models are item-based CF (IBCF) [4] and weighted-regularized matrix factorization (WR-MF) [7, 15] respectively. We will describe these models in Section 5. Another popular implicit feedback MF model is Bayesian Personalized Ranking (BPR) [19] that uses pairwise comparison of items by assuming that the items on which the user interacted must be ranked higher than the items on which the user did not.

Recommendations can also be generated by employing data mining techniques. For example, Mobasher et al. [13] use pattern mining methods to discover sequential patterns of user actions. The patterns are used for predicting on which items the user will act. These items are considered as good recommendations. A recommender based on Markov chains was studied by Zimdars et al. [23]. Frequent pattern mining and statistical models have been used in music playlist generation techniques [2].

As we mentioned in the introduction, the interactions between a user and an item may not be limited to a single type of actions (e.g., only clicks or ratings). It has been recently shown that it is possible to improve a target action prediction by leveraging information about multiple action types (clicks, views, purchases, etc.) [18, 17, 5]. Another line of research is investigating the usage of time information such as the delays between actions, actions' recency, or the time spent on browsing items [5, 10, 22].

Existing models can also be combined in more complex, and supposedly more effective hybrid recommender systems [3]. The hybrid system proposed by Rendle et al. (FPMC) [20] is very related to that proposed in this paper, as it combines SM and CF techniques, as we have done. However, their system uses CF (tensor factorization with BPR optimization criteria) to learn personalized Markov chains (the SM model), while we use an SM model to generate input for a CF predictor. A drawback of FPMC is that it is not designed for handling multiple action types, which is a peculiar and novel feature of our proposed approach.

Another hybrid system that is similar to ours was introduced by Melville et al. [12]. The Content-Boosted Collaborative Filtering (CBCF) hybrid approach incorporates Content-based (CB) and CF models to provide a user with movie recommendations. Differently from our system, CBCF exploits both user and item features. The CB component uses these features to densify the original user-item ratings matrix by predicting the unknown user-item ratings. Then, the densified matrix is used by the CF component to generate recommendations.

**Table 1: A sample of the interactions data**

| *user_id* | *item_id* | *interaction_type* | *created_at* |
|---|---|---|---|
| 1974005 | 2668706 | 1 | 1444154047 |
| 2690450 | 405777 | 2 | 1445338496 |
| 2690450 | 1180447 | 3 | 1444806365 |
| 2690450 | 1981683 | 4 | 1444894156 |

## 3. DATASET

The dataset provided by XING[2], a career-oriented social networking platform, for the ACM RecSys Challenge 2016 has been used for the empirical evaluation of the proposed models. This dataset records interactions performed by users on job postings. A sample of the data is presented in Table 1. Actually, this dataset is a semi-synthetic sample of an original XING data, and it is enriched with artificial users whose presence contributes to the anonymization. Moreover, some noise was added to the data: not all interactions of a user are contained in the dataset while some of the interactions are artificial (have actually not been performed by the user)[3].

A row in the table represents an action/interaction of type *interaction_type* performed by the user *user_id* on the job posting (item) *item_id* at time *created_at*. There are four types of actions: 1 – the user *clicked* on the item, 2 – the user *bookmarked* the item, 3 – the user clicked on the *reply button* or *application form button* and 4 – the user deleted the recommendation from her recommendations list.

Our hybrid system is designed to predict whether a user will perform an action of a target type on an item, and we assume that "reply" is the target action. However, the pro-

[1]http://2016.recsyschallenge.com

[2]https://www.xing.com

[3]https://github.com/recsyschallenge/2016/blob/master/TrainingDataset.md

**Table 2: Statistics of the used data**

| | |
|---|---|
| # of users | 784687 |
| # of items | 1029480 |
| # of actions | 8826678 |
| % of reply actions | 4.8 |
| % of delete actions | 11.5 |
| Avg. # of actions performed by a user | 11.3 |

posed approach can also be used for predicting other action types.

# 4. DATA REPRESENTATION

As it was previously mentioned, the proposed hybrid recommender system integrates SM and CF prediction models. The SM model predicts the probability that a user will perform an action of a target type on an item, given the information of the user's previous interactions with the item. Similarly to Ben-Shimon et al. [1], we define the chronologically ordered sequence of all the actions (of any type) performed by a user on an item as a user-item *session*. Hence, a user-item pair is associated to one session only, but, it is possible that a user-item session contains several actions of the same type. For example, if a user replied to a job posting three times then the corresponding session will contain three reply actions.

We denote with $a^j_{ui}$ the $j^{th}$ action performed by the user $u$ on the item $i$, and by $s_{ui} =< a^1_{ui}, a^2_{ui}, \ldots, a^n_{ui} >$ a user-item session. We refer to this representation as a *timeless* session. *Time-aware* sessions are instead represented as $< a^1_{ui}, d^1_{ui}, \ldots, a^n_{ui}, d^n_{ui} >$, where $d^j_{ui}$ is the observed time delay (e.g., in seconds) between action $a^j_{ui}$ and $a^{j+1}_{ui}$. The last time delay $d^n_{ui}$ is the time passed after the last action $a^n_{ui}$ was performed by $u$ on $i$. This information is required for more accurate estimation of the probability of observing the target action at the next, $(n+1)^{th}$, position in the session.

To fit the proposed SM models, sessions must be compared. Comparing timeless sessions is quite straightforward; for example, two sessions can be compared element-wise or by considering their bi-gram representations. However, time-aware sessions require more complex approaches. In that respect, we decided to discretize a continuous time delay into three levels: $s$ - short, $m$ - medium and $l$ - long. For instance, the time-aware session $< click, 5, bookmark, 15, reply, 60 >$ is discretized as $< click, s, bookmark, m, reply, l >$. In general, the number of intervals may vary depending on the data and the domain.

To identify the endpoints of the discrete intervals we analyzed the distribution of the time delays observed in all the available sessions (Fig. 1). We found that the distribution is exponential. The 0.33 and 0.66 quantiles of this distribution have been used as separating values between the $s$, $m$ and $l$ intervals. Thus, the time delays lying in the intervals $[0, 0.33q)$ and $[0.33q, 0.66q)$ are considered as $s$ and $m$, while time delays greater or equal to $0.66q$ are considered as $l$.

# 5. PREDICTION MODELS

In this section we describe the proposed hybrid recommender system leveraging multiple action types. We introduce the components of the system and the method we used to combine them.

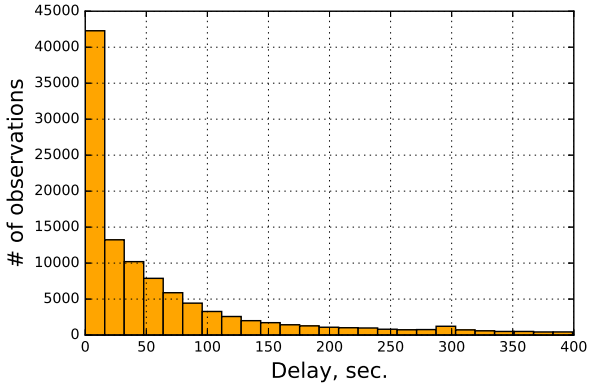


**Figure 1: The distribution of the time delays between actions of all users. The largest delay is 400 seconds.**

## 5.1 SM Models

The first component of the proposed hybrid system predicts, for all the sessions that are not ending with the target action $a^T$, the probability that the next unobserved user action will be the target. In this paper we introduce several alternative probabilistic models, which we denote with $\Theta$, and compare them with each other. All the models require a learning step that is performed according to Algorithm 1. $P(a^T|s_{ui})$ is the probability that in a session that is not ending with the target action the user will perform next the target action.

**Algorithm 1** Target action prob. estimation procedure

```
for all session s in a training set do
    # split s to prefix and outcome parts
    outcome ← the last action in s
    prefix ← the prefix of s until the outcome

    # define the key as a transformation of the prefix part
    # (transformation function depends on the model)
    key ← T(prefix)

    # using the key and the outcome update the model
    update()
end for
```

During the learning process each session is split into *prefix* and *outcome*. For example, the session $< 1, s, 2, m, 3, m >$, where the action types are represented by digits, is split into the prefix $< 1, s, 2, m >$ and the outcome 3. Depending on the model, the prefix is transformed into a *key*. For instance, a transformation function that converts sessions into sets of symbols transforms the prefix $< 1, m, 1, m >$ into the key $\{1, m\}$. The outcome, i.e., the ending of the session, is used as ground-truth. Together, the key and the outcome, are used to learn a $\Theta$-model which estimates the conditional probability $P(a^T|s_{ui})$.

We introduce here some useful notations to simplify the explanation. We denote with $N(k)$ the function that returns the number of sessions in the training set containing the key value $k$. The function $N_T(k)$ returns the number of sessions in the training set that contain the key value $k$ and end

with $a^T$. The value $N$ returns the number of sessions in the training set, while $N_T$ is the number of sessions in the training set ending with $a^T$. The number of keys learned by a model is $K$.

**SeqSet:** This target action probability prediction model employs a transformation function $T$ that converts an input session $s_{ui}$ into a set of symbols $k_{ui}$ (i.e., a set of distinct actions and delays constituting $s_{ui}$).

$$T(< 1, s, 2, m, 2, m >) = \{1, 2, m, s\}$$

The model estimates the conditional probability as:

$$P(a^T|s_{ui}) = P(a^T|k_{ui}) = \begin{cases} \frac{N_T(k_{ui})}{N(k_{ui})} & if\ N(k_{ui}) > 0 \\ \frac{1}{K} & otherwise \end{cases}$$

**Bayes:** The $T$ function of the Bayes model transforms an input session $s_{ui}$ into a list of suffixes arranged in descending order according to the size of a suffix.

$$T([1, s, 2, m, 3, l]) = [[1, s, 2, m, 3, l], [2, m, 3, l], [3, l], []]$$

The idea behind such a transformation is that a user can perform the target action after any of the sub-histories of her interaction with a given item. During the training process, every suffix is used to train the model. However, the conditional probability is found only for the longest suffix $l_{ui}$ for which $N(l_{ui}) > 0$. A similar procedure of shortening the initial sequence was proposed by Mobasher et al. [13]. The model estimates the conditional probability by applying Bayes rule:

$$P(a^T|s_{ui}) = P(a^T|l_{ui}) = \frac{P(l_{ui}|a^T)P(a^T)}{P(l_{ui}|a^T)P(a^T) + P(l_{ui}|!a^T)P(!a^T)}$$

$$P(a^T|l_{ui}) = \frac{\frac{N_T(l_{ui})}{N_T}\frac{N_T}{N}}{\frac{N_T(l_{ui})}{N_T}\frac{N_T}{N} + \frac{N_{!T}(l_{ui})}{N_{!T}}\frac{N_{!T}}{N}}$$

where $!a^T$ is an action not of the target type and $N_{!T}$ is the number of sessions that do not end with the target type of action.

**Naive Bayes:** The Naive Bayes (NB) model assumes that actions in a session are mutually independent, conditioned to the fact that the last action in $s_{ui}$ is $a^T$ (or $!a^T$ in the case of $P(s_{ui}|!a^T)$). This is equivalent to assume that the input session is described by the list of its composing actions (action-delay pairs in the case of time-aware sessions).

$$T([1, s, 2, m, 3, l]) = [[1, s], [2, m], [3, l]]$$

Let us denote with $s_{ui}[j]$ the $j^{th}$ action (or action-delay pair) of the session $s_{ui}$. During the training process, all composing parts of the sessions are used to train the model. The conditional probability is estimated by applying the Bayes rule:

$$P(a^T|s_{ui}) = \frac{P(s_{ui}|a^T)P(a^T)}{P(s_{ui}|a^T)P(a^T) + P(s_{ui}|!a^T)P(!a^T)}$$

However, because of actions independency we have:

$$P(s_{ui}|a^T) = P(s_{ui}[1]|a^T) \cdots P(s_{ui}[n]|a^T)$$

By applying smoothing we also have:

$$P(s_{ui}[j]|a^T) = \frac{N_T(s_{ui}[j]) + 1}{N(s_{ui}[j]) + K}$$

## 5.2 Collaborative Filtering Models

In this section, we introduce two CF models that we will compare with the proposed hybrid model.

### 5.2.1 Item-Based Collaborative Filtering

Item-Based Collaborative Filtering (IBCF) is a neighborhood model leveraging relationships between items that can be observed in the user-item interaction data. In IBCF, the unknown user-item interactions are predicted by aggregating interaction data of the same user with items similar to the target one. Two item-to-item similarity measures are frequently used: Pearson correlation coefficient and cosine similarity. They are computed based on a user-item interaction matrix $R$, where $r_{ui} = 1$ if $u$ performed the target action on $i$ (e.g., $r_{ui} = 1$ if $u$ replied on $i$). A learning procedure is executed to generate predictions $\hat{r}_{ui}$ for the items $i$ where $r_{ui}$ is unknown. Using a similarity measure, one can identify the set of $k$ items $I^k(u, i)$, on which $u$ acted, that are most similar to $i$. If we denote with $sim_{ij}$ the similarity between items $i$ and $j$, then an unknown interaction is estimated as:

$$\hat{r}_{ui} = \frac{\sum_{j \in I^k(u,i)} sim_{ij} r_{uj}}{\sum_{j \in I^k(u,i)} sim_{ij}}$$

To improve IBCF prediction accuracy we used some techniques described in [4]. First, we normalized each row of the $R$ matrix to be of unit length. Then, we computed the similarity between items using the cosine function. The similarities were also normalized to be of unit length.

### 5.2.2 Weighted-Regularized Matrix Factorization

Weighted-Regularized Matrix Factorization (WR-MF) is a state-of-the-art implicit feedback CF model. The input data for the model is a user-item non-negative matrix $A$, whose entries $a_{ui}$ count the number of observed actions of $u$ on $i$. To predict whether a user will act on an item the *indicator* and *confidence* values are introduced. The indicator value $p_{ui}$ shows whether the user $u$ has performed the target action on the item $i$ ($p_{ui} = 1$) or not ($p_{ui} = 0$). During the training phase $p_{ui} = 1$ if $a_{ui} > 0$, otherwise $p_{ui} = 0$. If $p_{ui} = 1$, we are confident on this information, as it is based on real observations of the user $u$ acting on the item $i$ ($a_{ui} > 0$). Conversely, the model should not be confident that $u$ will not act on $i$ in the future when $a_{ui} = 0$. To model the confidence, WR-MF computes, on the base of $a_{ui}$, a confidence score $c_{ui}$ for each $p_{ui}$ entry. The state-of-the-art choice for confidence estimation is $c_{ui} = \alpha * a_{ui} + 1$, where $\alpha \geq 1$ is a meta-parameter that must be optimized.

The target action prediction is then computed using MF: each user $u$ and item $i$ is associated with an $f$-dimensional factors vector $x_u \in \mathbb{R}^f$ and $y_i \in \mathbb{R}^f$ respectively. The predicted value of the indicator function is computed by the dot product of these two vectors: $\hat{p}_{ui} = x_u^T y_i$. The vectors' parameters are learned by minimizing the following cost function:

$$\min_{x*,y*} \sum_{u,i} c_{ui}(p_{ui} - x_u^T y_i)^2 + \lambda(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2)$$

The constant $\lambda$ is used for tuning the amount of regularization and avoid model's overfitting [16]. Cost function minimization is achieved by alternating least squares (ALS) optimization [7]. The values of $\alpha$ and $\lambda$ are data-dependent and determined by cross-validation (described in Section 6).

In our experiments, setting $\alpha = 40$ and $\lambda = 0.001$ was found to produce optimal results.

## 5.3 Hybrid Model

In the proposed hybrid approach the model learned by a Sequence Mining (SM) technique is then used by a Collaborative Filtering (CF) technique to produce the final action predictions and recommendations. We use the notation SM + CF to indicate the hybridization. During the training phase an SM model $\Theta$ generates a user-item matrix $M$, where $m_{ui}$ is given only if the corresponding session $s_{ui}$ exists. $m_{ui} = P(a^T|s_{ui}, \Theta)$ if the session does not end with the target action and it is 1 if the observed last action is the target. Then a CF model employs the matrix $M$ as input to predict entries of the matrix $M$ that are either unknown or not equal to 1. According to Burke [3], the proposed hybrid recommender model can be considered as a meta-level hybrid: "the model learned by one recommender is used as input to another". In the rest of this paper we have studied the performance of two hybrid models: SM + IBCF and SM + WR-MF. In our experiments, the WR-MF component of SM + WR-MF produces optimal result with $\alpha = 80$ and $\lambda = 0.001$.

# 6. EVALUATION PROCEDURE

## 6.1 Training and Testing Sets

The chronological order of the actions is crucial in the considered application. Hence, in order to split the data into training and testing sets we used a time-dependent splitting criteria. The first 80% time stamped data records were used for the training, while the remaining 20% was used as the testing set. Being split by a time point, a user-item session may fall partially in the training and partially in the test. In this case, the first part of the session is considered as a part of the training set, while the remaining part stays in the testing set.

For the models that require optimization of meta-parameters (i.e., WR-MF, SM + WR-MF) the training set was split into training (80%) and validation parts (20%). The meta-parameters were optimized by benchmarking the trained model on the validation part. As soon as the meta-parameters were identified the models were retrained on the original training set. Thus, the testing set has never been used by the models throughout the training phase.

During the training phase of the SM models the available sessions are partitioned into two groups: those ending with an action of a target type ($a^T$) and those ending with another action type ($!a^T$). The non-target action types are clicks, bookmarks, and delete. But, while the delete action is alternative to the reply action, i.e., no item is both "deleted" and "replied", click and bookmark may coexist (on the same item). Therefore, to train the models to clearly distinguish between positive and negative outcomes we left in the training set only sessions ending with either the reply or the delete actions.

To guarantee the functionality of IBCF we kept in the dataset only users and items that have in the training set at least two sessions ending with the reply action. Hence, we considered only users that replied to at least two job postings and postings that received at least two replies from users. Additionally, we removed from the testing set users and items that were not present in the training set.

## 6.2 Protocol & Metrics

During the evaluation, for each user in the testing set a model generates a list of items (recommendations) which are predicted to receive a reply by the user. Because it is more interesting for a user to be recommended with items that she has not recently acted, or that she is not aware of [7], we have removed from the recommendations all the items for which a user has any sessions ending with reply action in the training set. Then, we select top-$k$ recommended items, i.e., the items with the largest estimated probability.

We note that the goal of the ACM RecSys Challenge 2016 was to predict those job postings that a user will have an interaction different from "delete" (i.e., either click or bookmark or reply). Though, in principle, the proposed hybrid model can be used to predict such a generically positive interaction, we did not focus on this particular task, hence we cannot compare our results with those obtained in the ACM RecSys Challenge 2016.

We have assessed the quality of the proposed models by measuring Mean Average Precision at $k$ (MAP@k) and F1-score at $k$ (F1@k) [11]. These metrics are based on the precision and recall metrics that we define in the following. Let us denote with $M_u$ the set of top-$k$ ranked recommended items for a user $u$ and with $T_u$ the set of items in the test data to which the user $u$ replied. If $U^T$ is a set of test users, then Precision at k (P@k) and Recall at k (R@k) can be defined as follows:

$$P@k = \frac{1}{|U^T|} \sum_{u \in U^T} \frac{|M_u \cap T_u|}{k} \quad R@k = \frac{1}{|U^T|} \sum_{u \in U^T} \frac{|M_u \cap T_u|}{|T_u|}$$

MAP is the arithmetic mean of average precision values across recall levels over test users. If $J_u$ is the set of the positions of the relevant items for a user $u$ in $M_u$, then:

$$MAP@k = \frac{1}{|U^T|} \sum_{u}^{U^T} \frac{1}{|J_u|} \sum_{j}^{J_u} P@j$$

F1-score is the harmonic mean of precision and recall, thus:

$$F1@k = \frac{2 * P@k * R@k}{P@k * R@k}$$

In the evaluation experiments $k \in \{1, 3, 5, 10\}$, since in real-world applications only the very first recommended items are typically browsed by users.

# 7. RESULTS

We first compare the stand-alone SM and CF models. Descriptive statistics of the training and testing sets are shown in Table 3. Tables 4 and 5 show MAP@k and F1@k of the models trained on time-aware sessions. Because of lack of space we do not show the performance of the models trained on timeless sessions: overall, they perform on average 5% worse than those built on time-aware sessions.

There are only small differences between the performances of the various SM models described in Section 5.1. However, the best MAP@k and F1@k are obtained by Bayes. Besides, among the CF model the best performance is obtained by IBCF. We also note that the best performances of the MF based models are obtained with 200 latent factors, and with a training process running for 10 iterations.

As we have mentioned previously, we have considered two variants of the proposed hybrid models: SM + IBCF and

**Table 3: Statistics of the training and testing sets**

| | |
|---|---|
| # of session ending with reply in the training set | 37815 |
| # of session ending with delete in the training set | 25591 |
| # of session ending with reply in the testing set | 1338 |
| # of users | 10822 |
| # of items | 12250 |
| Max density of the user-item training matrix | 0.1% |

**Table 4: Models comparison in terms of MAP@k. Best values are underlined.**

| Approach | Model | @1 | @3 | @5 | @10 |
|---|---|---|---|---|---|
| SM | SeqSet | 0.033 | 0.043 | 0.048 | 0.052 |
| | Bayes | 0.034 | 0.042 | 0.048 | 0.052 |
| | NB | 0.025 | 0.040 | 0.045 | 0.049 |
| CF | IBCF | 0.031 | 0.035 | 0.043 | 0.047 |
| | WR-MF | 0.023 | 0.025 | 0.031 | 0.038 |
| SM + CF | NB + IBCF | <u>0.051</u> | <u>0.055</u> | <u>0.063</u> | <u>0.073</u> |
| | NB + WR-MF | 0.036 | 0.046 | 0.052 | 0.060 |

SM + WR-MF. In order to simplify the assessment of the performance of SM + IBCF and SM + WR-MF, we show in Tables 4 and 5 only the performance obtained by the best combinations of the SM and CF components.

The hybrid model NB + IBCF outperforms both the best SM model (Bayes) and the best stand-alone CF model (IBCF). This is especially noticeable for MAP@k where the improvement is at least 30%. Also, the NB + WR-MF model outperforms the plain SM and CF models, but the improvement is not as large as that obtained by NB + IBCF. We believe that the weak model in NB + WR-MF is WR-MF. The degree of sparseness of input matrices $A$ and $M$ is conjectured to be the reason of such low accuracy of WR-MF.

We finally note that the performance difference between SM + CF and the stand-alone SM and CF models, for $k \in \{3, 5, 10\}$, is statistically significant with $p < 0.01$ (t-test). To perform the significance test we run the experiments five times using only 90% of data for the training and testing purposes, each time with a different random seed.

# 8. CONCLUSIONS

In this paper, we have introduced a hybrid user action prediction model that uses Sequence Mining (SM) and Collaborative Filtering (CF). The proposed model leverages only information about the users' performed actions at a certain point in time and predicts whether a target user will perform an unobserved action of a target type on an item. The model does not employ any user and item content-related information. The empirical evaluation, which was conducted on a large real-world dataset, has shown that the proposed hybrid model outperforms stand-alone SM and CF.

**Table 5: Models comparison in terms of F1@k. Best values are underlined.**

| Approach | Model | @1 | @3 | @5 | @10 |
|---|---|---|---|---|---|
| SM | SeqSet | 0.027 | 0.037 | 0.038 | 0.035 |
| | Bayes | 0.028 | 0.036 | 0.040 | 0.035 |
| | NB | 0.021 | 0.036 | 0.038 | 0.035 |
| CF | IBCF | 0.022 | 0.037 | 0.036 | 0.026 |
| | WR-MF | 0.017 | 0.020 | 0.024 | 0.025 |
| SM + CF | NB + IBCF | <u>0.038</u> | <u>0.048</u> | <u>0.045</u> | <u>0.038</u> |
| | NB + WR-MF | 0.026 | 0.032 | 0.031 | 0.028 |

The proposed hybrid model is more general than FPMC [20] and can incorporate as subcomponents any SM and CF model. Our model exploits the ordering of the actions, the time delays between them, and can be adapted to make use of multiple action types. Furthermore, due to the modular nature of the model, any improvement in the SM or CF components can be easily adopted to build more accurate predictions.

The proposed hybrid model has some limitations related to the nature of its components. First of all, it can work only with user-item interactions presented in the form of sequences of actions. Next, the currently implemented SM component does not exploit potential relationships between actions of different types, such as interaction effect or the impact of one action to another. As a result, in case of rare sequences the probability estimations performed by the SM component can be inaccurate. Finally, like any other CF system, the proposed hybrid model suffers from the cold-start problem.

For future work, we plan to compare the proposed hybrid model with FPMC and CBCF, and to test it on other datasets. Additionally, we would like to extend the model by incorporating information about user and item features, which we believe may help to cope with the cold-start problem.

The source code of all models and experiments is available at https://gitlab.inf.unibz.it/tural-gurbanov/probability_hybrid.

# 9. REFERENCES


[1] D. Ben-Shimon, A. Tsikinovsky, M. Friedmann, B. Shapira, L. Rokach, and J. Hoerle. Recsys challenge 2015 and the yoochoose dataset. In *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, pages 357–358, New York, NY, USA, 2015. ACM.

[2] G. Bonnin and D. Jannach. Automated generation of music playlists: Survey and experiments. *ACM Comput. Surv.*, 47(2):26:1–26:35, Nov. 2014.

[3] R. Burke. The adaptive web. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web*, chapter Hybrid Web Recommender Systems, pages 377–408. Springer-Verlag, Berlin, Heidelberg, 2007.

[4] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177, Jan. 2004.

[5] T. Gurbanov, F. Ricci, and M. Ploner. Modeling and predicting user actions in recommender systems. In *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization*, UMAP '16, pages 151–155, New York, NY, USA, 2016. ACM.

[6] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 230–237, New York, NY, USA, 1999. ACM.

[7] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings*

*of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 263–272, Washington, DC, USA, 2008. IEEE Computer Society.

[8] D. Jannach, L. Lerche, and M. Jugovac. Adaptation and evaluation of recommendations for short-term shopping goals. In *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, pages 211–218, New York, NY, USA, 2015. ACM.

[9] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, Aug. 2009.

[10] L. Lerche and D. Jannach. Using graded implicit feedback for bayesian personalized ranking. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, pages 353–356, New York, NY, USA, 2014. ACM.

[11] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[12] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Eighteenth National Conference on Artificial Intelligence*, pages 187–192, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[13] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Using sequential and non-sequential patterns in predictive web usage mining tasks. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, pages 669–672, 2002.

[14] D. Oard and J. Kim. Implicit feedback for recommender systems. In *in Proceedings of the AAAI Workshop on Recommender Systems*, pages 81–83, 1998.

[15] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *2008 Eighth IEEE International Conference on Data Mining*, pages 502–511, Dec 2008.

[16] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. *Proceedings of KDD Cup and Workshop*, pages 39–42, 2007.

[17] L. Peska and P. Vojtas. Using implicit preference relations to improve recommender systems. *Journal on Data Semantics*, pages 1–16, 2016.

[18] H. Qiu, G. Guo, J. Zhang, Z. Sun, H. T. Nguyen, and Y. Liu. Tbpr: Trinity preference based bayesian personalized ranking for multivariate implicit feedback. In *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization*, UMAP '16, pages 305–306, New York, NY, USA, 2016. ACM.

[19] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, pages 452–461, Arlington, Virginia, United States, 2009. AUAI Press.

[20] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 811–820, New York, NY, USA, 2010. ACM.

[21] F. Ricci, L. Rokach, and B. Shapira. *Recommender Systems Handbook*, chapter Recommender Systems: Introduction and Challenges, pages 1–34. Springer US, Boston, MA, 2nd edition, 2015.

[22] X. Yi, L. Hong, E. Zhong, N. N. Liu, and S. Rajan. Beyond clicks: Dwell time for personalization. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, pages 113–120, New York, NY, USA, 2014. ACM.

[23] A. Zimdars, D. M. Chickering, and C. Meek. Using temporal data for making recommendations. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, UAI '01, pages 580–588, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.