# Collaborative Filtering with Behavioral Models

Dušan Sovilj
University of Toronto
Toronto, Ontario
dusans@mie.utoronto.ca

Scott Sanner
University of Toronto
Toronto, Ontario
ssanner@mie.utoronto.ca

Harold Soh
National University of Singapore
Singapore, Singapore
harold@comp.nus.edu.sg

Hanze Li
University of Toronto
Toronto, Ontario
litos.li@mail.utoronto.ca

## ABSTRACT

Collaborative filtering (CF) has made it possible to build personalized recommendation models leveraging the collective data of large user groups, albeit with prescribed models that cannot easily leverage the existence of known behavioral models in particular settings. In this paper, we facilitate the combination of CF with existing behavioral models by introducing Bayesian Behavioral Collaborative Filtering (BBCF). BBCF works by embedding arbitrary (black-box) probabilistic models of human behavior in a latent variable Bayesian framework capable of collectively leveraging behavioral models trained on all users for personalized recommendation. There are three key advantages of BBCF compared to traditional CF and non-CF methods: (1) BBCF can leverage highly specialized behavioral models for specific CF use cases that may outperform existing generic models used in standard CF, (2) the behavioral models used in BBCF may offer enhanced intepretability and explainability compared to generic CF methods, and (3) compared to non-CF methods that would train a behavioral model per specific user and thus may suffer when individual user data is limited, BBCF leverages the data of all users thus enabling strong performance across the data availability spectrum including the near cold-start case. Experimentally, we compare BBCF to individual and global behavioral models as well as CF techniques; our evaluation domains span sequential and non-sequential tasks with a range of behavioral models for individual users, tasks, or goal-oriented behavior. Our results demonstrate that BBCF is competitive if not better than existing methods while still offering the interpretability and explainability benefits intrinsic to many behavioral models.

## CCS CONCEPTS

• **Computing methodologies** → **Learning in probabilistic graphical models**; **Mixture models**; **Learning latent representations**;

## KEYWORDS

collaborative filtering; behavioral modeling; bayesian model averaging

**ACM Reference Format:**
Dušan Sovilj, Scott Sanner, Harold Soh, and Hanze Li. 2018. Collaborative Filtering with Behavioral Models. In *UMAP '18: 26th Conference on User Modeling, Adaptation and Personalization, July 8–11, 2018, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3209219.3209235


*UMAP'18, July 8–11, 2018, Singapore, Singapore*

ACM ISBN 978-1-4503-5589-6/18/07…$15.00
https://doi.org/10.1145/3209219.3209235

## 1 INTRODUCTION

Collaborative filtering (CF) methods [8] for personalized recommendation leverage data from multiple users, under the basic assumption that some users share similar behavior (preferences, actions) and thus one's behavior may be predicted by leveraging observations of others' behaviors. CF methods have become popular in recent years due to their performance and success in various competitions (e.g., the Netflix Challenge). Typical CF methods can be split into two basic variants [4]: (a) memory-based methods (e.g, k-Nearest Neighbor methods [20]) that use similarity functions between users or items to produce predictions, and (b) model-based methods (e.g., matrix/tensor factorizers [19] or deep learning methods [12, 21]) that apply machine-learning techniques to learn latent factors that best describe the observed data.

While highly successful, one caveat of existing personalized CF recommenders is that they prescribe their own models of behavior using generic machine learning methodologies such as nearest neighbors or latent embeddings inherent to matrix/tensor and deep models. However, in this paper, we ask how one can start with an individual application-specific behavioral model germane to a particular recommendation setting and leverage this behavioral model in a more general CF framework? The answer we provide comes in the form of Bayesian Behavioral Collaborative Filtering (BBCF), which works by embedding arbitrary (black-box) probabilistic models of human behavior in a latent variable Bayesian graphical model framework. Through principles of Bayesian inference, BBCF learns to recommend for a given user by leveraging a vote of the prediction of each behavioral model weighted according to the posterior probability that it could have generated the user's observed behavioral history. Critically for the behavioral modeling foundations of BBCF, users may explore a large space with very few overlapping preferences or actions, yet evidence of common behavioral patterns may still suggest strong similarity for recommendation purposes.

BBCF provides three key advantages compared to traditional CF and non-CF methods: (1) BBCF can leverage highly specialized behavioral models for specific use cases that may outperform existing application-independent models used in standard CF; (2) the behavioral models used in BBCF may offer enhanced intepretability and explainability compared to generic CF methods (i.e., a recommendation is now a weighted combination of these behavioral models); and (3) compared to non-CF methods that would train a behavioral model per specific user and thus may suffer when individual user data is limited, BBCF leverages the data of all users thus enabling strong performance across the data availability spectrum including the near cold-start case.

We evaluate BBCF on a range of datasets covering collaborative content-based movie tagging, adaptive user interface behavior prediction, navigation choice recommendation, and educational tutoring, where we demonstrate broad applicability of our method. Experimentally, we compare BBCF to individual and global behavioral models as well as CF techniques; our evaluation domains span both sequential and non-sequential tasks with a range of behavioral models for individual users, tasks, or goal-oriented behavior. Our overall results demonstrate that BBCF is competitive if not better than existing state-of-the-art methods. In short, BBCF provides a highly general collaborative filtering methodology for building personalized recommender systems from known behavioral models while enjoying the interpretability and explainability benefits of these underlying behavioral models.

## 2 RELATED WORK

We are not the first to suggest collaborative filtering through inferred behavioral similarity, however we strictly generalize the range of applicability of existing frameworks. An early proposal for leveraging behavioral similarity in collaborative filtering was provided by Personality Diagnosis (PD) [17], but it focused only on simple Gaussian rating vectors. In this work, we generalize this approach to arbitrary graphical user models and do not require common item sets or pre-agreed rating or label meanings. More recently, the robotics and controls communities have focused on leveraging multi-user data in intent-aware navigation, gesture, and other goal-oriented human action prediction models [2, 3, 5, 18]. This work did not identify connections nor applications to collaborative filtering nor compare to such methods as we do in this paper; further we will show that one can instantiate our framework for this specific goal-oriented setting as we demonstrate empirically on the Taxi trajectory prediction task, but our framework is strictly more general and intended for a broader range of CF applications.

## 3 BAYESIAN BEHAVIORAL COLLABORATIVE FILTERING

### 3.1 General Framework

In this section, we provide a probabilistic derivation of Bayesian Behavioral Collaborative Filtering (BBCF), where we assume the existence of a known class of pretrained behavioral models. These behavioral models, in principle, capture a broad range of models aimed at tackling a variety of user-oriented data, including but not limited to individual user behaviors, task-defined behaviors, and goal-driven behaviors.

We begin with a general modeling framework for BBCF, then provide a variant for the special case of temporal data. Graphical models for both formulations of BBCF are given in Figure 1.

Let $b \in B$ be a specific behavior label drawn from a discrete set $B$. Given the data generated by $b$ denoted as $\mathcal{D}_b = \{(x_i, y_i)\}_{i=1}^{|\mathcal{D}_b|}$, an individualized behavioral model $M_b$ can be trained with $\mathcal{D}_b$ to predict $y_i$ given $x_i$. Specifically, we assume that training the model results in a learned parameter vector $\theta_b$ and that the model produces predictive distributions $p(y|x, M_b) = p(y|x, \theta_b)$.

Now, assume we have a collection of behavioral models $\mathcal{M} = \{M_b\}_{b=1}^{|B|}$. Each behavioral model $M_b$ is trained independently and has associated parameters $\theta_b$ obtained using $\mathcal{D}_b$. Similar to the framework in [6, 14], we assume we have some historical data $\mathcal{D}_r$ for a user (with unknown behavior) for whom we wish to make a future prediction. We define a data-dependent weight vector $\mathbf{w}(\mathcal{D}_r) = [w_b(\mathcal{D}_r)]_{b=1}^{|B|} \in \mathcal{W}$, where $\sum_b w_b(\mathcal{D}_r) = 1$, and our prediction,

$$y_*(x_*; \mathcal{D}_r) = \sum_{b=1}^{|B|} w_b(\mathcal{D}_r) y_b(x_*), \tag{1}$$

which is a convex combination of individual behavioral model predictions $y_b(x_*)$.

But what form should the weights $\mathbf{w}$ take[1]? Under the assumption that a "sufficiently good" model exists in $\mathcal{M}$, we argue that $\mathbf{w}$ is a one-hot binary vector. To elaborate, when $r$'s model is in this collection, $M_r \in \mathcal{M}$, under typical losses (e.g., squared loss) and consistent estimators, the asymptotically optimal weight is a one-hot binary vector with 1 at the index of $r$. In the non-limiting case, we make the basic assumption underlying collaborative filtering methods: that users are not completely distinct and share similar behaviors. Specifically, we assume that there exists another model $M_a \in \mathcal{M}$ with $\theta_a$ sufficiently close to $\theta_r^*$, making it a good proxy for the true model.

In both cases, there is one "correct" model responsible for generating the observed data and thus, we limit $\mathcal{W}$ to the space of one-hot binary vectors $\mathcal{W} = \{\mathbf{w} \in \{0, 1\}^{|B|} : \sum_t w_b = 1\}$ and place a categorical distribution over $\mathbf{w}$.

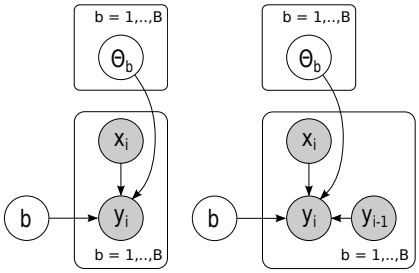
Here, the distribution over the averaged prediction $y_*(x)$ is

$$p(y_*|x, \mathbf{w}, \mathcal{D}_r, \mathcal{M}) = \sum_{b=1}^{|B|} w_b(\mathcal{D}_r) p(y_*|x, M_b), \tag{2}$$

which we can recognize as a finite mixture model with $\mathbf{w}$ acting as the latent "membership" variable; $w_b(\mathcal{D}_r)$ selects the appropriate model $M_b$ given the observed data $\mathcal{D}_r$. Marginalizing (2) over the probability of $\mathbf{w} = [w_b]$ yields:

$$\begin{aligned} p(y_*|x, \mathcal{D}_r, \mathcal{M}) &= \sum_{b=1}^{|B|} p(w_b = 1|\mathcal{D}_r) p(y_*|x, M_b) \\ &= \sum_{b=1}^{|B|} p(M_b|\mathcal{D}_r) p(y_*|x, M_b) \end{aligned} \tag{3}$$

[1] We omit denoting $\mathbf{w}$'s dependence on $\mathcal{D}_r$ to simplify the exposition.




(a) Nonsequential (b) Sequential

**Figure 1: Graphical representations for the individual behavior-centric data. For both cases we have $B$ underlying behavioral patterns each parameterized by specific instantiations $\theta_b$. (a) The general BBCF model where behavioral context $x_i$ in conjunction with behavioral model $b$ generates observed behavior $y_i$. (b) The special case of BBCF for Markovian temporal models where the behavioral context $x_i$, previous observed behavior $y_{i-1}$, and behavioral model $b$ generates current observed behavior $y_i$.**

since $p(w_b = 1|\mathcal{D}_r) = p(M_b|\mathcal{D}_r)$, and

$$p(M_b|\mathcal{D}_r) = p(\mathcal{D}_r|M_b)p(M_b)/p(\mathcal{D}_r) = \frac{\prod_i p(y_i|x_i, M_b)p(M_b)}{\sum_{b'} \prod_i p(y_i|x_i, M_{b'})p(M_{b'})}. \quad (4)$$

Equation (3) with (4) can be identified as Bayesian Model Averaging (BMA) applied over a collection of behavioral models.

## 3.2 General BBCF

As an example of Figure 1(a) for the general case of BBCF, we assume data originates from a group of users, where each behavioral model $b$ is defined per user and produces tag $y_i$ given item context $x_i$. Given a new user, we want to find other users with similar tagging styles. Here, we see that a distribution over latent tagging behavior model $b$ would be inferred for the new user based on their tagging history and the tagging predictions of each model $b$ for new context $x_i$ would be averaged according to this distribution to produce tag prediction $y_i$.

## 3.3 Sequential BBCF

In the special case of temporal or sequential data provided in Figure 1(b), we again assume data originates from many users, where each behavioral model $b$ is defined per user (or task or goal, as appropriate) and produces observed behavior $y_i$ given context $x_i$ and previous observed behavior $y_{i-1}$. Note that context $x_i$ can be empty. To make this concrete, we consider two examples from our experiments. For an adaptive user interface model, we define a behavior $b$ per user task where interface context $x_i$, previous action $y_{i-1}$, and task $b$ combine to predict the next interface action $y_i$. For navigation recommendation, we define a behavior $b$ per navigation goal where driving context $x_i$, goal $b$, and previous location $y_{i-1}$ combine to predict the next recommended location $y_i$.

# 4 BASELINE MODELS

## 4.1 Generative Models

We first cover two generative probabilistic models for non-sequential and sequential data respectively that can be incorporated as behavioral models for use in BBCF. In conjunction with application-specific feature representations, these models represent two of many possible probabilistic behavioral models for use in BBCF.

*4.1.1 Naïve Bayes.* For non-sequential data where samples $x_i$ are feature vectors containing features $x_{ij}$, we can use the Naïve Bayes as our baseline behavioral model used to predict $P(y_i|x_i) \propto P(y_i) \prod_j P(x_i j|y_i, b)$ for each behavior $b$.

*4.1.2 (Hidden) Markov model.* A common approach to time series modeling for a sequence of observations is to assume that $y_i$ is generated not only from the current behavior $b$ and context $x_i$, but also from $y_{i-1}$ generated in the immediately preceding time step. The modeling of such Markov Models requires the estimation of a stationary distribution $P(y_i|y_{i-1}, b, x_i) = P(y_{i-1}|y_{i-2}, b, x_{i-1}), \forall i$. When $x_i$ and $y_i$ are finite discrete random variables, $P(y_i|y_{i-1}, b, x_i)$ may be defined in terms of a well-known transition matrix $T_{b,x_i} : y_{i-1} \times y_i \rightarrow [0, 1]$.

A Hidden Markov Model (HMM) assumes additional structure wherein the Markovian transition structure is not directly over the observable $y_i$, but instead over a latent state variable $z_i$ that generates $y_i$. The HMM requires a transition model over latent state $P(z_i|z_{i-1}, b, x_i)$ along with an additional observation model $P(y_i|z_i)$ representing a generalization of the provided sequential Markov model.

## 4.2 Recommendation Baselines

In addition to the previous generative models for BBCF, we also provide standard baselines for recommendation that will be compared to in this paper.

*4.2.1 Nearest neighbors.* $k$ nearest-neighbor methods ($k$NN) represent one of the most common types of recommender system [1]. For the extension of $k$NN methods to sequential data, we transform sequences into frequency counts between adjacent pairs of states; for a problem with $N$ states, each sequence is therefore transformed into an $N \times N$ count matrix (or $N^2$ length vector).

We use two standard distance functions to match observable behavior: cosine and Euclidean distance. Given the $k$ nearest neighbors, we predict or recommend $y_i$ as the majority vote among the $y_i$ predicted by the neighbors with ties broken randomly. The number of neighbors $k$ is chosen via cross-validation with $k \in \{1, \ldots, 10\}$.

*4.2.2 LSTM Recurrent Neural Network.* Specifically for sequential data, Long-Short Term Memory (LSTM) [13] based neural networks have gained recent popularity as a strong baseline recommendation model [12]. Our models use a single layer of LSTM cells with training data composed of sequences of prespecified length. The network learns to predict the next state by producing the probabilities for all states (via softmax layer). Hyperparameters of the model (learning rate, number of neurons in LSTM cell, batch size during training phase, dropout rate) are obtained via 3-fold cross validation on available data. Once the hyperparameter values are selected, the final training phase runs for a maximum of 50 epochs

**Table 1: Summary of sequential data sets**

| | #samples | | seq. length | | |
|---|---|---|---|---|---|
| Data | train | test | min | max | avg |
| synthetic | 13500 | 1500 | 13 | 50 | 32.0 |
| Taxi | 64881 | 7209 | 2 | 48 | 10.3 |
| UI | 89 | 8 | 43 | 572 | 185.8 |
| KT-SK1 | 141 | 15 | 17 | 115 | 36.6 |
| KT-SK2 | 168 | 29 | 16 | 195 | 68.4 |
| KT-SK3 | 116 | 12 | 17 | 96 | 38.8 |
| KT-SK4 | 505 | 56 | 7 | 213 | 50.8 |

with an early stopping criterion defined on a separate validation set to prevent overfitting.

## 5 EXPERIMENTAL DOMAINS

To test the proposed BBCF method against commonly used approaches, we use four data sets: one synthetic, three sequential real-world data sets, and one collaborative tagging data set. Table 1 summarizes all sequential data sets.

### 5.1 Synthetic Sequential Prediction

The rationale for synthetic data construction is to vary the degree of separation between different behaviors to observe how this affects BBCF and other recommenders. Synthetic data is constructed by first specifying several transition models/matrices $T^k$ by hand and the uniform transition matrix $T^{\mathrm{U}}$. We introduce a separation level $\lambda$ to indicate distinction between the individual models $T^k$ and the uniform model $T^{\mathrm{U}}$ as $\bar{T}^k = \lambda T^k + (1-\lambda)T^{\mathrm{U}}$. With $\lambda = 1$ all transition matrices are intact and sufficiently different, while $\lambda = 0$ squashes all transitions to a single uniform model. Varying $\lambda$ provides some intermediate level between two extremes. For each $\lambda$, we generate up to $N$ samples for each task $T^k$ ending with a total of $K * N$ samples. We have chosen $K = 3$ tasks, each containing 4 states and sampled 5000 points for each task ending with 15k data sequences in total. The lengths of sequences varied between 5 and 50.

### 5.2 Taxi Navigation Recommendation

This data set was provided as part of a competition affiliated with ECML/PKDD in 2015 entitled *The Taxi Service Trip Time* competition [16], but we are interested in recommending the next position given a particular prior travel path of a taxi.

The trajectories of taxi cars are monitored via GPS from the initial state to their goal state and the locations are given in (longitude, latitude) pairs over a period of time. In order to simplify calculations, all sequences have been converted to fall within a grid of prespecified size.

Unusually long sequences have been removed prior to removing self-transitions since they heavily dominate the data. For our experiments, we chose first 100k samples of the training data file, and a grid map of size $20 \times 20$, giving us roughly 70k samples.

### 5.3 Adaptive User Interface Recommendation

The third sequential data set is constructed from logs of an experiment involving an adaptive user interface (UI). The participants were asked to perform two tasks involving communication network (of users and emails) while the logger recorded their actions. The two tasks were set up as follows: (1) identify the source user who issued a *malicious* email, and (2) label as many nodes that satisfy certain conditions. More details about the interface itself are given in [15]. There are total of 14 actions possible within the interface and the problem is predicting the next action the user will perform.

The number of available samples in this domain is very low due to the limited number of human participants observed in this dataset. For task (1) there are 76 available samples and 13 samples for task (2). For our experiments we use 4 samples from each task to form the test set.

### 5.4 Assistments Tutoring Prediction

Knowledge Tracing (KT) models are designed to capture the student learning process by inferring latent knowledge states and future student performance. In this domain, Bayesian knowledge tracing (BKT) remains one of the most applied models due to its relative performance and interpretability. One downside of the standard BKT method is the need to train per-user (individual) models when the interaction data is limited [7]. On the other hand, pooling all available data in order to train a single (global) model ignores the fact that each student's learning characteristics can vary substantially and that individualized models provide better accuracy [22].

Here, we use Assistments dataset [9] that we further categorize into four skill sets: Charts & Graphs, Probability & Statistics, Angles and Fractions, where each skill set consists of several conceptually related skills. For each skill, we observe a sequence of binary problem outcomes indicating whether the student answered each problem correctly. The summary statistics given in Table 1 involve *pooling* all users together across their assignments for each chosen skill set.

### 5.5 MovieLens Collaborative Tagging

Collaborative tagging (CT) systems (e.g., MovieLens, Flickr) allow users to "tag" content (e.g., movies, documents, photos) with keywords, thus providing socially-procured metadata for exploration, search and retrieval [10]. Often, CT systems provide assistance to users in the form of recommended tags that can be suggested based on tagging history and resource content.

We evaluate the non-sequential aspect of our method on the MovieLens dataset [11], which contains approximately 580k tags applied by 247k users across 34k movies. Similar to prior studies (e.g., [19]), we work with a 10-core dataset (movies with at least 10 tags, with users that have tagged at least 10 movies), leaving 46k tagging events by 1198 users on 1597 movies using 198 tags. To generate movie content features, we transformed movie synopses obtained from the IMDB database into normalized 5000-word count histograms $\mathbf{x}_j \in \mathbb{R}^{5000}$.

## 6 EXPERIMENTAL RESULTS

In our experiments[2], we aim to evaluate the performance of BBCF under a wide variety of behavioral models on real datasets: one

[2]Code is available at https://github.com/dusansovilj/umap18_bbcf

non-sequential classification behavior model (MovieLens Collaborative Tagging), and on the sequential side, one *user-based* behavioral model (Assistments Tutoring Prediction), one *goal-oriented* behavioral model (Taxi Navigation Recommendation), and one *task-oriented* behavioral model (Adaptive User Interface Recommendation). We include an additional sequential behavioral model (Synthetic) to explore how BBCF and other baseline systems perform as the similarity between different behaviors varies from complete behavioral overlap to complete independence (since this can be directly modulated for synthetic data).

For baseline methods, we use (Hidden) Markov model, nearest-neighbor and Long-Short Term Memory (LSTM) neural nets in the case of sequential data, and the Naïve Bayes classifier for feature-based data. We omit the nearest-neighbor and neural network based models for the non-sequential collaborative tagging case as both require a common set of items between different users to be tagged, which does not hold in our experiments (i.e., our objective is to learn tagging behavior conditioned on document content, not to recommend specific tags for specific documents).

First, we aim to address two important questions regarding our approach for both non-sequential and temporal data:

1. How does BBCF compare to non-collaborative filtering baselines using per-user trained **individual** behavioral models and a single **global** behavioral model that pools all data?
2. How does BBCF compare to standard recommendation baselines?

Outcomes from these first two questions served as guidelines for the next set of questions targeting the sequential domains only:

3. How well does each method perform in a variable task separation scenario?
4. How does the number of samples impact overall performance?
5. How does the performance change as prediction horizon increases?

## 6.1 Baseline comparisons

*6.1.1 Non-sequential (tagging) case.* In this experiment, we combine individualized content-based tag recommenders using the BBCF framework. Our base user content-based classifier is a generative model, where given resource $j$ with content $\mathbf{x}_j$ and user-tag parameters $\theta_u = \{\theta_{u,i}\}_{i \in \mathcal{I}}$, the probability that tag $i \in \mathcal{I}$ is applied by user $u$ is given by,

$$p(y_{i,u,j}|\mathbf{x}_j, \theta_u) = \frac{1}{Z} p(\mathbf{x}_j|y_{i,u,j}, \theta_{u,i}) p(y_{i,u,j}) = \frac{1}{Z} \prod_k^K p(x_{j,k}|y_{i,u,j}, \theta_{u,i}) p(y_{i,u,j}) \quad (5)$$

where $y_{i,u,j} \in \{0, 1\}$, $Z = \sum_{i,j} p(\mathbf{x}_j|y_{i,u,j}, \theta_{u,i}) p(y_{i,u,j})$ is the normalization factor, and we have assumed that the $K$ content features factorize given the tag (Naïve Bayes assumption). To obtain a recommendation, we marginalize over the parameters (obtained by training each user model separately),

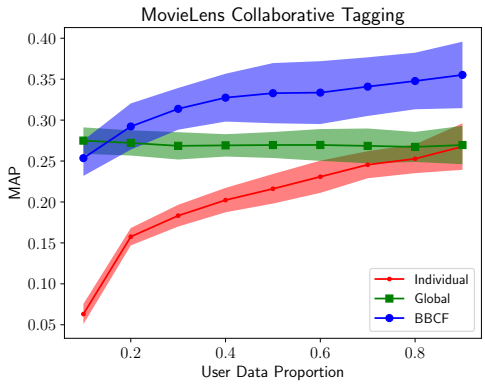



**Figure 2: Mean Average Precision (MAP) for the three approaches: global, individual and BBCF framework on MovieLens data. Global model is trained on *full* data, while individual models are trained per user data set $\mathcal{D}_r$.**

$$p(y_{i,v,j}|\mathbf{x}_j, \{\theta_u\}_u) = \sum_u p(y_{i,u,j}|\mathbf{x}_j, \theta_u) p(\theta_u|\mathcal{D}_v) \quad (6)$$

where $\mathcal{D}_u = \{(\mathbf{x}_j, \hat{a}_{i,u,j})_{l=1}^L\}$ is a set of observed tags by the user $u$. The user model posterior given a likelihood function $\ell(\cdot)$ is

$$p(\theta_u|\mathcal{D}_v) = \frac{1}{Z} \prod_l^L \sum_{y_{i,v,j}} \ell(\hat{a}_{i,v,j}|y_{i,v,j}) p(y_{i,v,j}|\mathbf{x}_j, \theta_u) p(\theta_u). \quad (7)$$

In the following, we used a Bernoulli likelihood,

$$\ell(\hat{y}_{i,v,j}|y_{i,u,j}, \beta) = \begin{cases} \beta & \text{if } \hat{y}_{i,v,j} = y_{i,v,j} \\ 1 - \beta & \text{otherwise} \end{cases} \quad (8)$$

with observation noise parameter $\beta = 0.3$, and multinomial likelihoods for $p(y_{i,u,j}|\mathbf{x}_j, \theta_u)$.

We conducted 10-fold cross-validation where 80% of the users were used for training the models and the remaining 20% for testing. Model performance was measured using mean average precision (MAP):

$$\text{MAP} = \frac{1}{L} \sum_{l=1}^L \sum_{r=1}^R \text{Prec}(r) \Delta\text{Recall}(r) \quad (9)$$

where $r$ is the cut-off rank, $R$ is the number of recommended tags, $\text{Prec}(r)$ is the precision at the cut-off $r$, and $\Delta\text{Recall}(r)$ is the change in recall. For each test user, we measured the MAP for different proportions of user data to evaluate performance under varying test data volume conditions.

Figure 2 illustrates the MAP scores achieved by the three models as varying proportions of user tags from 10-90% were revealed. The global model was largely unaffected by the additional tags (MAP scores $\sim 0.27$) since the newly observed tags consisted small proportions of the overall data. As expected, the MAP scores increased as more tags were revealed for both the individual (0.06-0.27) and BBCF (0.25-0.35) models. The BBCF model clearly outperformed

**Table 2: Accuracy of prediction (with 95% confidence intervals in brackets) across four skill categories in Assistments data.**

| method | KT-SK1 | KT-SK2 | KT-SK3 | KT-SK4 |
|---|---|---|---|---|
| BBCF | 79.30 (1.82) | **72.71** (1.19) | 69.06 (1.63) | 78.15 (0.66) |
| HMM global | 78.24 (1.37) | 67.44 (0.81) | 66.42 (1.62) | 74.37 (0.48) |
| HMM individual | 74.37 (2.22) | 66.95 (1.55) | 63.82 (2.44) | 74.83 (0.72) |
| NN(cos) global | 70.40 (1.79) | 57.26 (1.88) | 55.38 (1.94) | 65.92 (1.39) |
| NN(cos) individual | 63.26 (1.52) | 55.06 (0.76) | 50.44 (1.33) | 65.45 (0.70) |
| NN(euc) global | 70.12 (2.40) | 57.26 (1.88) | 57.41 (1.51) | 67.62 (1.31) |
| NN(euc) individual | 63.26 (1.52) | 55.06 (0.76) | 50.44 (1.33) | 65.45 (0.70) |
| LSTM global | **88.32** (3.09) | 70.32 (4.13) | **78.52** (5.11) | 79.24 (1.28) |
| LSTM individual | 87.88 (3.75) | 69.05 (4.73) | 56.76 (8.29) | **79.52** (4.07) |

the individual model at all proportions, indicating that movie tagging behaviors were not entirely distinct between users. Although not directly observed here, it is expected that the individual model would eventually "catch up" to BBCF as more data is provided.

*6.1.2 Sequential cases.* For the initial testing case, we use Assistments Tutoring data where we performed 10-fold cross validation with the following scheme: 80% of the users were used to train individualized models and testing was performed on the remaining 20% of the users. At each test iteration, the models were used to predict the attempted problem outcome, and after each sequence, the test user's model was updated with the observations. To enhance performance in the global model setting, the global model was re-trained with the newly observed test sequence included in the training set since it is computationally viable to incrementally retrain a single global model.

Table 2 shows the performance across four extracted skill sets. The outcome heavily depends on the specific skill category, with some being easier to predict than others. Overall, the LSTM provides the best results (both in global and individualized aspects), while our BBCF method is close to LSTM performance in two cases (KT-SK2 and KT-SK3). Compared to other baselines, BBCF is able to surpass both the Markov model and nearest-neighbor methods. Another gain over LSTMs is reduced variance as tuning LSTM hyperparameters is noisy and depends on the quality of available training data. BBCF also supports explainability and interpretability regarding how students are similar to each other via their learning behavior provided as their posterior model weights over $b \in B$.

Given that individualized models are slightly inferior to their global counterparts for both MovieLens and Assistments data, we resort to testing only single global model for the rest of the experiments. In addition to accuracy, we also measure performance in terms of Hit Rate at 2 (HR@2), that is, among the two highest rated predictions, is there one that is correct. The results are obtained via 10-fold cross-validation procedure.

Table 3 shows the performance of all methods on the remaining sequential data sets with results averaged across complete test sequences. Overall, the proposed BBCF outperforms the other methods and in some cases by quite a margin (Synthetic and Taxi data) which can be attributed to incorporating historical (and behavioral) information into the approach. In the case of synthetic data, we see that if "exact" models are in the pool of behavioral models $B$, BBCF can easily identify the underlying pattern and provide substantially better results than other methods. On the other hand, results on

**Table 3: Performance measurements (with 95% confidence intervals in brackets) across three data sets tested over complete sequences. HR@1 is the same as accuracy.**

| Data | method | HR@1 | HR@2 |
|---|---|---|---|
| Synthetic | BBCF | **84.10** (4.85) | **96.15** (1.49) |
| | HMM | 43.93 (10.72) | 67.09 (3.53) |
| | NN(cos) | 36.64 (3.72) | 37.72 (2.18) |
| | NN(euc) | 36.71 (3.68) | 37.80 (2.15) |
| | LSTM | 39.01 (1.71) | 70.70 (4.48) |
| Taxi | BBCF | **64.80** (0.03) | **85.01** (0.20) |
| | HMM | 44.37 (0.02) | 72.62 (0.24) |
| | NN(cos) | 28.45 (0.03) | 45.26 (0.24) |
| | NN(euc) | 28.74 (0.04) | 44.65 (0.36) |
| | LSTM | 47.84 (0.05) | 68.98 (0.71) |
| User Interface | BBCF | 52.13 (0.06) | 70.59 (0.71) |
| | HMM | 46.60 (0.02) | 64.80 (0.72) |
| | NN(cos) | 20.01 (0.13) | 20.01 (0.13) |
| | NN(euc) | 36.37 (0.03) | 36.37 (0.03) |
| | LSTM | **54.22** (0.08) | **74.38** (1.00) |

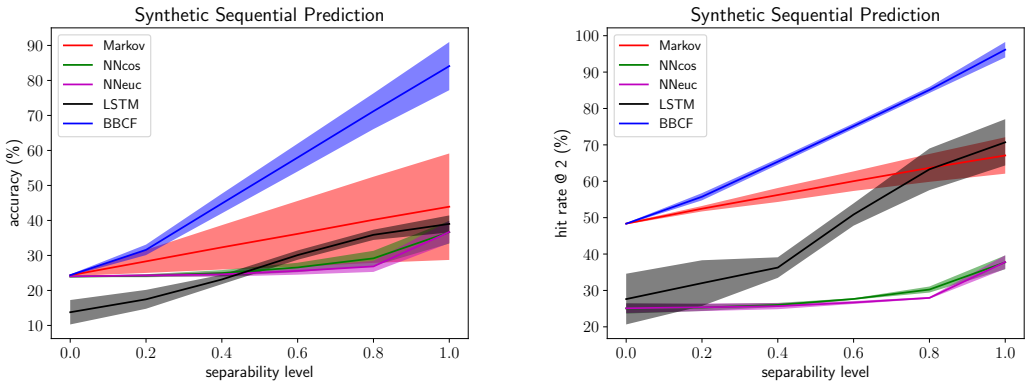
Adaptive User Interface indicate low numbers of samples lead to similar predictions for Markov and BBCF , while the superior performance of neural nets might signify that the tasks in this context are not easily separated and behavioral patterns are more similar than for the Taxi domain, which has considerably more goal states.

In both sequential and non-sequential domains, we are able to increase our prediction accuracy when compared to the baseline model (for both individualized and global modeling approaches) and still provide comparable performance to the more advanced (but less intepretable and explainable) models. Stronger behavioral generative models may also further reduce the gap.

## 6.2 Varying conditions for sequential data

For this set of experiments, we measure both accuracy and hit rate at 2 and report average performance and confidence intervals via 10-fold cross-validation.

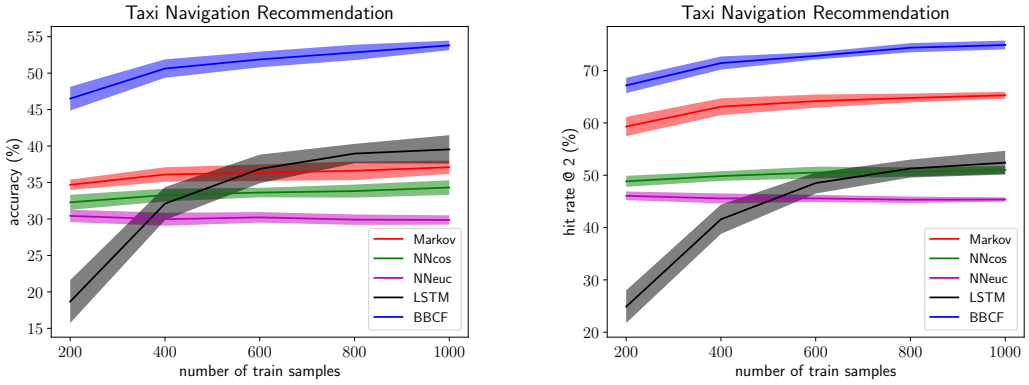
For the Markov model, all trajectories are taken into account to compute the transition matrix of probabilities $T$. In order to make all transitions possible from a particular state $s$ to all its adjacent states (given the constraints in the domain), a small Laplace smoothing prior of 0.1 is added to frequency counts. For BBCF , a Markov model is the individual model, where $M_b$ is constructed on sequences




(a) Hit Rate @ 1 (accuracy)

(b) Hit Rate @ 2

**Figure 3: Performance (and 95% confidence intervals) of methods under varying levels of separability between three tasks.**




(a) Hit Rate @ 1 (accuracy)

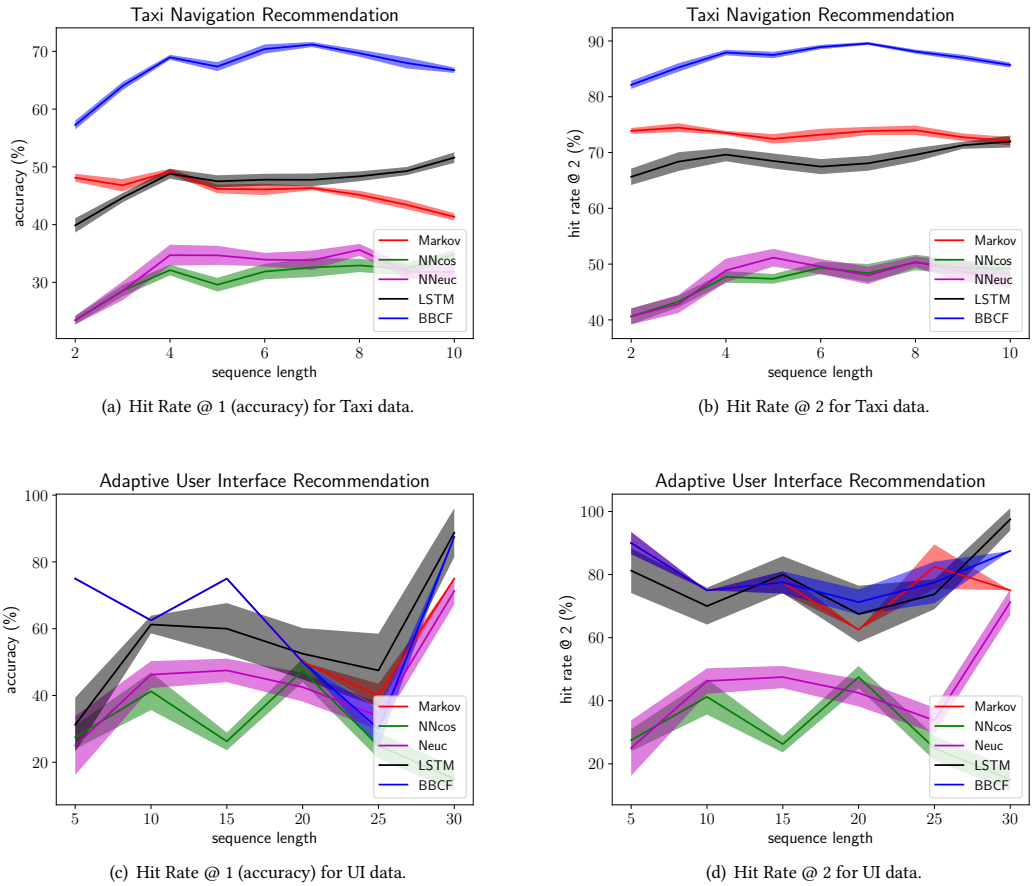
(b) Hit Rate @ 2

**Figure 4: Performance (and 95% confidence intervals) of methods under varying number of samples on the Taxi data.**

belonging to behavioral pattern $b$. For $k$NN, we use only the first neighbor (selected via cross-validation for all data sets) and we also take partial trajectories into account, that is, we introduce new samples that are shorter versions of original data. In this way, we ensure that sequences of similar lengths are used for matching neighbors.

*6.2.1 Varying task separability.* First, we wish to assess how well we are able to distinguish between different behavioral patterns associated with specific underlying tasks. We use the Synthetic data where we can easily control the level of separation between tasks by construction. Figure 3 shows the accuracy of all tested methods under different separability values. When the tasks are indistinguishable, all methods have very similar outcomes which is quite close to a random guess (given the domain with 4 states), but as the separation increases, BBCF is able to detect more precisely which tasks $y_*$ is responsible for the given trace $x_*$. In this setting, we are able to heavily outperform the more complex deep learning based LSTM baseline since BBCF's behavioral models are very accurate as the separability level increases and BBCF can easily identify the correct behavioral model for a user with high probability.

*6.2.2 Varying number of data samples.* In order to assess how fast BBCF can learn to recommend accurately, we vary the number of available training samples. For this experiment, we use the Taxi Navigation data set, but limit the domain to the 50 most frequently visited destinations and ensure that all goals are equally represented during the training phase. Figure 4 showcases the outcome for all methods. We see that BBCF is able to outperform other methods even in the low-sample scenario and the performance increases with more available data. The Markov model and NN do not benefit greatly from more data and either remain constant (NN) or slightly improve (Markov), partially because all necessary information is already present in the reduced case. The LSTM model greatly benefits from increased training data, but the performance is still far inferior to BBCF . In this goal-oriented scenario where users do not necessarily traverse similar paths and where there are only a few overlapping points, BBCF is able to extract the *intended behavior* for




(a) Hit Rate @ 1 (accuracy) for Taxi data.

(b) Hit Rate @ 2 for Taxi data.

(c) Hit Rate @ 1 (accuracy) for UI data.

(d) Hit Rate @ 2 for UI data.

**Figure 5: Performance (with 95% confidence intervals) of methods on sequential data given specific sequence length.**

new taxi customers by leveraging all encountered behaviors, while the LSTM appears unable to infer such goal-oriented behavior.

*6.2.3 Varying prediction horizon.* In our final experiment, Figure 5 shows recommendation performance with respect to test sequence length for the Taxi Navigation and Adaptive User Interface domains. We see that BBCF is increasingly able to model the correct posterior weighting over goal states with increasing sequence length on the Taxi data. This holds for several steps as we predict the general or initial direction for new traces. However, the performance (of all methods) drops slightly the longer the sequences become. The reason is that reaching the *exact* destination of a long test sequence may not be feasible since outlying destinations prevalent in longer sequences might not be present in the training data. Regardless, BBCF dominates all other methods.

The number of test cases for the Adaptive User Interface domain is considerably low (only 8) per segment so the variance is high for nearest-neighbor and LSTM, while Markov and BBCF perform comparably up to sequence length 20 then diverge. This divergence indicates that initial data traces up to length 20 (the average sequence length is 180) follow the same pattern and only beyond this point can BBCF identify a clear behavioral task separation.

## 7 CONCLUSION

We proposed the novel BBCF approach to collaborative filtering of behavioral models for personalized recommendation and prediction. This approach led to a convenient and efficient Bayesian model averaging solution for leveraging existing application-specific behavioral models. The BBCF framework is quite general as evidenced by its application to non-sequential collaborative tagging as well as sequential tasks with user-level, goal-oriented, and task-oriented behavioral models. Our results demonstrate that BBCF is competitive if not better than existing recommenders including deep learning models under a large variety of conditions (task separation, horizon length, and amount of data) while still offering the interpretability and explainability benefits intrinsic to many behavioral models.

## ACKNOWLEDGMENTS

The authors would like to thank Cong Shi for his help developing the initial codebase for some of the experiments. This work was partially supported by a NUS Office of the Deputy President (Research and Technology) Startup Grant.

# REFERENCES


[1] Robert M. Bell and Yehuda Koren. 2007. Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights. In *ICDM*. IEEE Computer Society, 43–52.

[2] Graeme Best and Robert Fitch. 2015. Bayesian intention inference for trajectory prediction with an unknown goal destination. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2015, Hamburg, Germany, September 28 - October 2, 2015*. 5817–5823. https://doi.org/10.1109/IROS.2015.7354203

[3] Graeme Best, Wolfram Martens, and Robert Fitch. 2017. Path Planning With Spatiotemporal Optimal Stopping for Stochastic Mission Monitoring. *IEEE Trans. Robotics* 33, 3 (2017), 629–646. https://doi.org/10.1109/TRO.2017.2653196

[4] John S Breese, David Heckerman, and Carl Kadie. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 43–52.

[5] Herbert Buchner, Karim Helwani, Bashar I. Ahmad, and Simon J. Godsill. 2017. Efficient adaptive filtering in compressive domains for sparse systems and relation to transform-domain adaptive filtering. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*. 3859–3863. https://doi.org/10.1109/ICASSP.2017.7952879

[6] Bertrand Clarke. 2003. Comparing Bayes Model Averaging and Stacking When Model Approximation Error Cannot Be Ignored. *Journal of Machine Learning Research* 4 (2003), 683–712. https://doi.org/10.1162/153244304773936090

[7] Albert T. Corbett and John R. Anderson. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. In *User Modelling and User-Adapted Interaction*, Vol. 4. 253–278. https://doi.org/10.1007/BF01099821

[8] Michael D. Ekstrand, John T. Riedl, and Joseph A. Konstan. 2007. Collaborative Filtering Recommender Systems. *Foundations and Trends in Human-Computer Interaction* 4321, 1 (2007), 291–324. https://doi.org/10.1504/IJEB.2004.004560 arXiv:ISSN 0018-9162

[9] Mingyu Feng, Neil T Heffernan, and Kenneth R Koedinger. 2006. Addressing the testing challenge with a web-based e-assessment system that tutors as it assesses. *Proceedings of the 15th international conference on World Wide Web - WWW '06* (2006), 307. https://doi.org/10.1145/1135777.1135825

[10] George W. Furnas, Caterina Fake, Luis von Ahn, Joshua Schachter, Scott Golder, Kevin Fox, Marc Davis, Cameron Marlow, and Mor Naaman. 2006. Why Do Tagging Systems Work?. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems (CHI EA '06)*. ACM, New York, NY, USA, 36–39. https://doi.org/10.1145/1125451.1125462

[11] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4, Article 19 (Dec. 2015), 19 pages. https://doi.org/10.1145/2827872

[12] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based Recommendations with Recurrent Neural Networks. *CoRR* abs/1511.06939 (2015).

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[14] A. Juditsky and A. Nemirovski. 2000. Functional aggregation for nonparametric regression. *Annals of Statistics* 28, 3 (2000), 681–712. https://doi.org/10.1214/aos/1015951994

[15] Sean W Kortschot, Dusan Sovilj, Harold Soh, Greg A Jamieson, Scott Sanner, Chelsea Carrasco, Scott Ralph, and Scott Langevin. 2017. An open source adaptive user interface for network monitoring. In *Systems, Man, and Cybernetics (SMC), 2017 IEEE International Conference on*. IEEE, 1535–1539.

[16] Luis Moreira-Matias, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. 2013. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems* 14, 3 (2013), 1393–1402.

[17] D M Pennock, E Horvitz, S Lawrence, and C L Giles. 2000. Collaborative filtering by personality diagnosis: A hybrid memory-and model-based approach. *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence* 64, 10 (2000), 473–480. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.38.6498{&}rep=rep1{&}type=pdf

[18] Eike Rehder and Horst Kloeden. 2015. Goal-Directed Pedestrian Prediction. *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)* 00 (2015), 139–147. https://doi.org/doi.ieeecomputersociety.org/10.1109/ICCVW.2015.28

[19] Steffen Rendle and Lars Schmidt-Thieme. 2010. Pairwise interaction tensor factorization for personalized tag recommendation. *Proceedings of the third ACM international conference on Web search and data mining (WSDM '10)* (2010), 81–90. https://doi.org/10.1145/1718487.1718498

[20] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. ACM, 175–186.

[21] S. Sedhain, A. Menon, S. Sanner, and L. Xie. 2015. AutoRec: Autoencoders Meet Collaborative Filtering. In *Proceedings of the 24th International Conference on the World Wide Web (WWW-15)*. Florence, Italy.

[22] Michael V. Yudelson, Kenneth R. Koedinger, and Geoffrey J. Gordon. 2013. Individualized bayesian knowledge tracing models. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7926 LNAI (2013), 171–180. https://doi.org/10.1007/978-3-642-39112-5-18