Project Report

Using collaborative filtering with frequent sequential patterns mining for product recommendation systems in multiple E-commerce Databases

Project Report by

RUTURAJ R. RAVAL

Professor

Dr. C. I. EZEIFE

Course

Emerging Non-Traditional Database System | 60-539

University of Windsor

Windsor, Ontario, Canada

Project Presentation on: April, 02nd 2018

Project Report submission on: April, 09th 2018

Table of Contents

1. Introduction

In everyday life, people rely on recommendations from other people, by spoken words, reference letters, news reports from news media, general surveys, travel guides, and so forth. Recommender system assists and augment this natural social process to help people sift through all available resources like, available books, articles, web pages, movies, music, restaurants, jokes, grocery products, etc. to find most interesting and valuable information for one such particular user that applies to end users. The developers of one of the first recommender systems, *Tapestry* [1] (traditional recommendation systems include rule-based recommenders and user customization), coined the phrase Collaborative Filtering (CF) [2]. Which has been widely adopted regardless of the facts that recommenders may not explicitly collaborate with recipients and recommendations may suggest particularly interesting items, in addition to indicating those that should be filtered out [3]. The fundamental assumption of CF is that if users *X* and *Y* rate *n* items similarly, or have similar behaviours (for an example buying, watching, listening, etc.) and hence will rate or act on other items similarly [4].

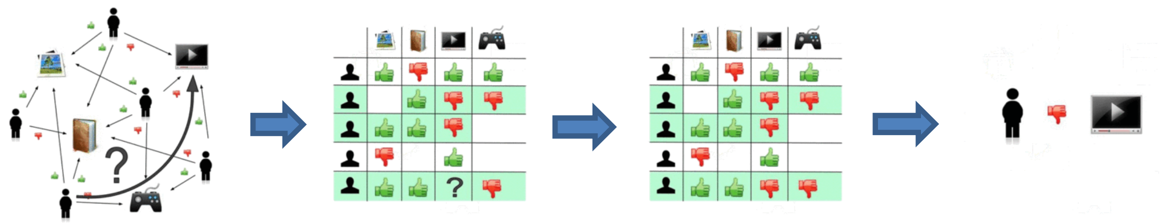
1.1 Collaborative Filtering

Collaborative Filtering (CF) is a technique used by the recommender system. CF has two senses- a narrow one and a more general one. The growth of the Internet has made it much more difficult to effectively extract useful information from all the available online information. The motivation for collaborative filtering comes from the idea that people often get the best recommendations from someone with tastes similar to themselves. Collaborative filtering encompasses techniques for matching people with similar interests and making recommendations on this basis. Collaborative filtering algorithms often require (1) users' active participation, (2) an easy way to represent users' interests, and (3) algorithms that are able to match people with similar interests [5]. Typically, the workflow of a collaborative filtering system is [5]:

- A user expresses his or her preferences by rating items (e.g. books, movies or CDs) of the system. These ratings can be viewed as an approximate representation of the user's interest in the corresponding domain.
- The system matches this user's ratings against other users' and finds the people with most "similar" tastes.

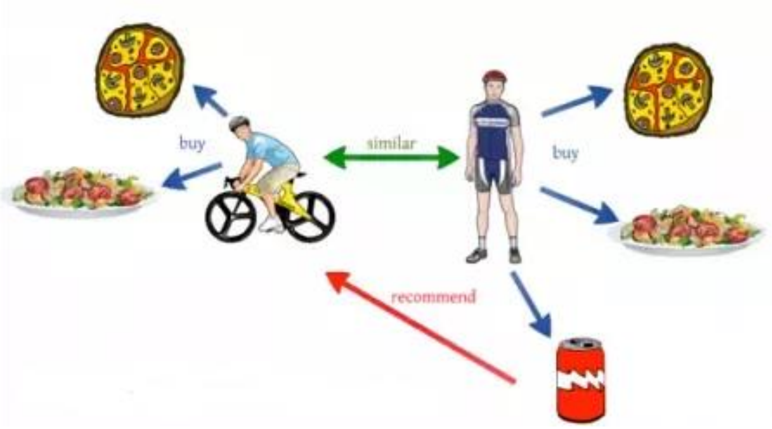
- With similar users, the system recommends items that the similar users have rated highly but not yet been rated by this user (presumably the absence of rating is often considered as the unfamiliarity of an item)

This set of images shows an example of predicting the user's rating using collaborative filtering. At first, people rate different items (like videos, images, games). After that, the system is making predictions about user's rating for an item, which the user hasn't rated yet. These predictions are built upon the existing ratings of other users, who have similar ratings with the active user. For instance, in this case, the system has made a prediction, that the active user won't like the video [5].



Collaborative filtering systems have many forms, but many common systems can be reduced to two steps:

1. Look for users who share the same rating patterns with the active user (the user whom the prediction is for).
2. Use the ratings from those like-minded users found in step 1 to calculate a prediction for the active user



This falls under the category of user-based collaborative filtering. A specific application of this is the user-based Nearest Neighbour algorithm.

Alternatively, item-based collaborative filtering (users who bought *x* also bought *y*), proceeds in an item-centric manner:

1. Build an item-item matrix determining relationships between pairs of items
2. Infer the tastes of the current user by examining the matrix and matching that user's data [5]

2. Problem statement

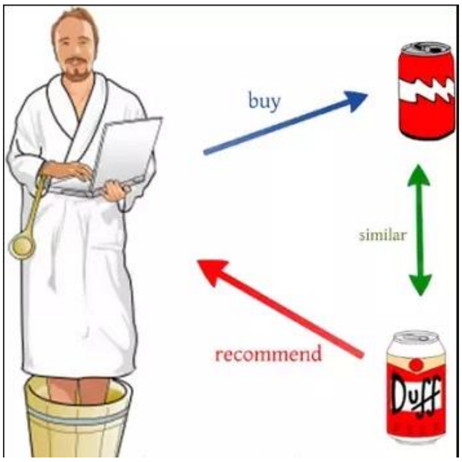
The traditional approach of recommending products to the users was not scalable enough due to different challenges of CF like, data sparsity, synonym problem, gray sheep, shilling attacks, diversity and long-tail, etc. therefore to solve the issues, researchers are working to make recommender system efficient enough in day-to-day life with the advancement of the technology.

3. Related work

The memory-based CF uses user rating data to compute the similarity between users or items. This is used for making recommendations. This was an early approach used in many commercial systems [5].

In model-based CF approach, models are developed using different data mining, machine learning algorithms to predict users' rating of unrated items. There are many model-based CF algorithms. Bayesian networks, clustering models, latent semantic models such as singular value decomposition, probabilistic latent semantic analysis, multiple multiplicative factor, latent Dirichlet allocation and Markov decision process based models [5].

The journey of recommenders system started with research papers on CF [6]. Recommendation systems are designed using various techniques including k-NN, decision tree, clustering, regression, heuristic methods, neural networks and association rule mining. Based on the type of techniques used, recommendation systems can be classified as content-based and collaborative based systems.



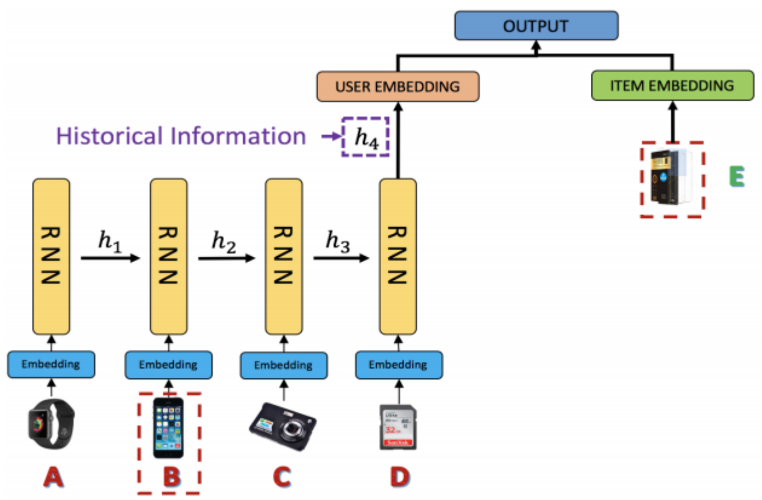
The content-based approach has originated from the information retrieval and information filtering domain [7]. Content-based recommender systems generate recommendations based on user's past preferences. The rating for any item for any user is calculated based on ratings of similar items given by the user. Many researchers treat it as a classification problem where the goal is to learn a function that predicts which class a document belongs to (either liked or non-liked) [8], others view it as a regression problem in which the goal is to learn a function that predicts a numeric value, i.e. the rating of the document [9] [10] [11].

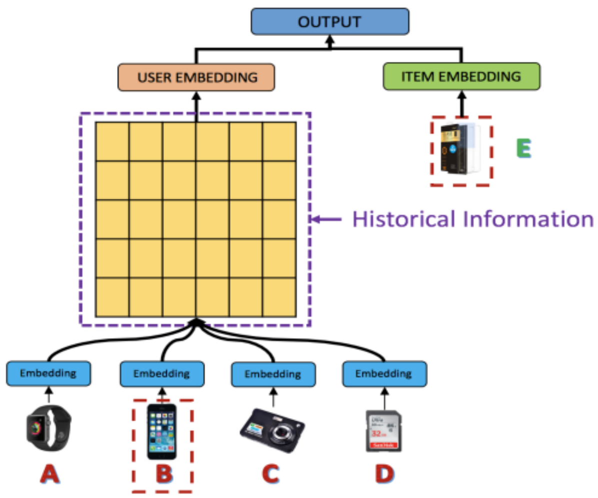
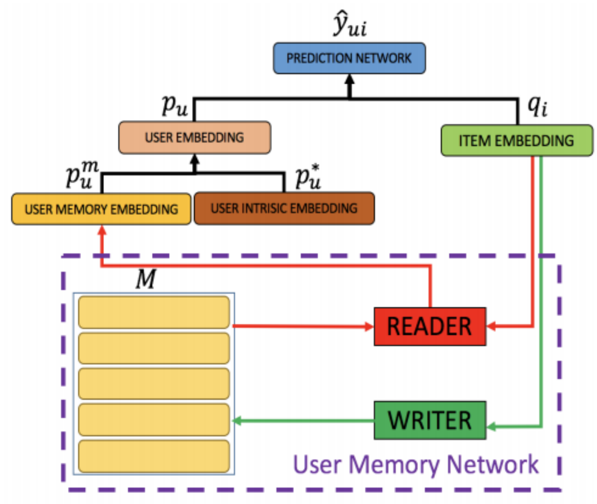
4. Proposed approaches

There are few effective and scalable and accurate algorithms which are there in the area of product recommendations in E-commerce using CF. I will be discussing four of such important algorithm which I think are helpful. Starting from RUM: Recommendations with the User-Memory network [12], secondly, Hybrid recommendation approach [13] [14], thirdly, Recommendations based on life-stage [15], etc.

4.1 Recommendations with the User-Memory network: RUM

The algorithm called *RUM: Recommendation with User Memory networks* will be taken as a reference. This algorithm works well to leverage external memory networks integrated with CF for sequential recommendation framework and provides the item-level and feature-level specifications of the framework.

	Sequential recommendations based on RNN ***Input***: An example of leveraging user historical records for recommendation. The user bought A, B, C and D in the past orderly. ***Middle-layer***: RNN ***Output***: these previous records (as embeddings of A, B, C and D) are summarized into h4 for predicting E.

<table>
<tr>
<td>
</td>
<td>General idea to adopt memory for recommendation
Input: The user bought A, B, C and D in the past orderly.
Middle-layer: Historical information stored in memory matrix
Output: Our general idea of introducing a per user memory matrix to store user historical records.</td>
</tr>
<tr>
<td>
</td>
<td>RUM general framework
Input: User memory network
Middle-layer: User and item embedding
Output: Prediction network</td>
</tr>
</table>

<u>Algorithm</u>:

i. To read the memory matrix,

   a. For an input q, first compute the similarity $S(q, m_i)$ between the input and the each memory slot m_i in the memory matrix and then the attention weights are derived by $w_i = SOFTMAX(S(q, m_i))$, where $SOFTMAX(xi) = \frac{e^{x_i}}{\sum_j e^{x_j}}$, based on which the memory is attentively read out.

ii. The personalized memory matrix M^u in an expressive structure,

   a. Specifically, for the user u, the memory embedding p_u^m is derived by reading M^u according to the current item embedding which was received from last section of currently similar item q_i. $\boldsymbol{p}_u^m = READ(\boldsymbol{M}^u, \boldsymbol{q}_i)$

b. And then by merging p_u^m with the user intrinsic embedding p_u^* we get the final user embedding as, $\boldsymbol{p}_u = MERGE(\boldsymbol{p}_u^*, \boldsymbol{p}_u^m)$

i. Where *MERGE(.)* is a function that merges two vectors into one. The particular choice of *MERGE(.)* in our model is a simple weighted vector addition, that is, $MERGE(x, y) \doteq x + \alpha y = \boldsymbol{p}_u^* + \alpha \boldsymbol{p}_u^m$

ii. Where α is a weighting parameter.

iii. Prediction function

a. When making predictions, we feed final user embedding p_u and the item embedding q_i into a function. $\hat{y}_{ui} = PREDICT(\boldsymbol{p}_u, \boldsymbol{q}_i)$

i. Where *PREDICT(.)* is an arbitrary prediction function or even a prediction neural network as in. Here, the sigmoid inner product is chosen, $\hat{y}_{ui} = \sigma(\boldsymbol{p}_u{}^T \cdot \boldsymbol{q}_i)$

iv. And at last, binary cross-entropy is adapted as loss function for model optimization, and the objective function to be optimized is,

$$
\begin{aligned}
l_{RUM} &= \log \prod_{(u,i)} (\hat{y}_{ui})^{y_{ui}} (1 - \hat{y}_{ui})^{1-y_{ui}} - \lambda ||\Theta||_F^2 \\
&= \sum_u \sum_{i \in I_u^+} \log \hat{y}_{ui} + \sum_u \sum_{i \in I / I_u^+} \log(1 - \hat{y}_{ui}) - \lambda ||\Theta||_F^2
\end{aligned}
$$

a. Where θ is the model parameter set, y_{ui} is the ground truth that would be *1* if u has purchased i, and *0* otherwise. I is the set of all items and I_u^+ is the set of items that u purchased arranged in purchasing order, namely, $I_u^+ = \{v_1^u, v_2^u, \cdots, v_{|I_u^+|}^u\}$

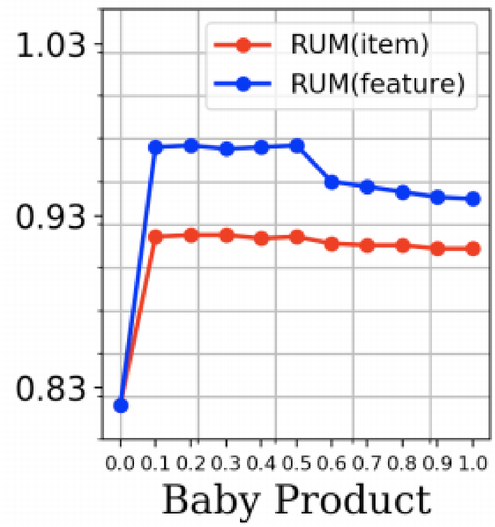
i. Where v_j^u is the j^{th} item purchased by u. we uniformly sample the negative instances from unobserved item set $I_u^- = I / I_u^+$

v. After each purchasing behaviour, we need to update the user memory matrix M^u to maintain its dynamic nature *b*, $\boldsymbol{M}^u \leftarrow WRITE(\boldsymbol{M}^u, \boldsymbol{q}_i)$

a. Where personalized memory matrix M^u encodes user behaviours and how to design the *READ(.)* and *WRITE(.)* on both item-levels and feature-levels.

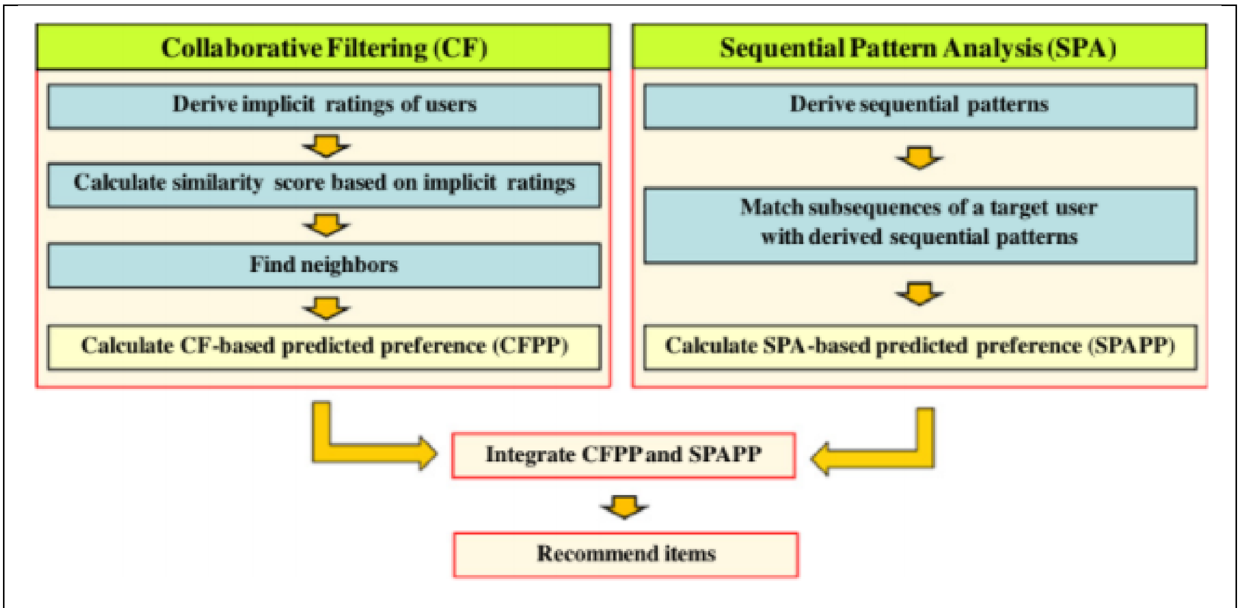
Example:

To illustrate intuition of item-to-item transitions example, which are sampled from the results of item-level RUM with 5 memory slots (K = 5) on the Baby Care dataset. When a user purchased her i-th item (corresponding to the x-axis, denoted in a blue grid), we plot a length-5 column vector in her subfigure (e.g., the boxed vector in the upper-middle subfigure), and the vector element corresponding to position j on the y-axis is the attention weight z, that this user casts on her j-th purchased item. When the user continues to purchase items, the most recently purchased 5 items are maintained in the memory, thus the column vectors shift from bottom-left to upper-right. The darker the color, the higher attention is casted on the item.



4.2 Hybrid recommendation approach

In hybrid approach, there is one algorithm called Hybrid Online Product rEcommendation (HOPE) system which integrates CF-based recommendations using implicit rating and SPA (Sequential pattern Analysis)-based recommendation. As shown in the image the overall framework of the recommendation system, which consists of two main processes: CF-process and SPA-process.



Overall framework of HOPE system

Input: Two main processes of CF and SPA

First middle-layer: Implicit rating provision in CF and calculate the similarity score, in SPA match the sub sequences of target user and derive sequential pattern from transaction data

Second middle-layer: Predicted preferences of a target user on items purchased by the top-k neighbours, CFPP are calculated and predicted preferences on items SPAPP are calculated

Output: weighted sum of the normalized CFPP and SPAPP is calculated as Final Predicted Preference: FPP, on each candidate item to recommend and the top-n items with highest FPP are recommended

Algorithm:

i. ***CF***: Derive implicit ratings of users on items

   a. The absolute preference of user u on item i, AP(u,i), is computed from the following equation. $AP(u,i) = \ln\left(\frac{\text{The number of transactions of user } u \text{ including item } i}{\text{The number of transactions of user } u} + 1\right)$

   b. The relative preference of user u on item i, RP(u,i) is thus defined as in the following equation. $RP(u,i) = \frac{AP(u,i)}{\underset{c \in U}{\text{Max}}(AP(c,i))}$

      i. where U denotes every user who purchased item i. Note that RP represents a user's preference to some extent, while AP does not.

   c. Finally, we multiplied RP(u,i) by 5 and rounded up so that implicit rating ranges from 1 to 5, as is mostly used in current recommendation systems, which is explained by the following equation. $\text{Implicit rating}(u,i) = \text{Round up } (5 \times RP(u,i))$

ii. Calculating similarity score based on implicit ratings

   a. $$\text{Pearson correlation coefficient}(a,b) = \frac{\sum_{i=1}^{m}(R_{ai} - \overline{R}_a)(R_{b,i} - \overline{R}_b)}{\sqrt{\sum_{i=1}^{m}(R_{a,i} - \overline{R}_a)^2}\sqrt{\sum_{i=1}^{m}(R_{b,i} - \overline{R}_b)^2}}$$

   $$\text{Cosine similarity}(a,b) = \frac{\sum_{i=1}^{m}(R_{a,i})(R_{b,i})}{\sqrt{\sum_{i=1}^{m}(R_{a,i})^2}\sqrt{\sum_{i=1}^{m}(R_{b,i})^2}}$$

   $$\text{Distance-based similarity}(a,b) = \frac{1}{1 + \sqrt{\sum_{i=1}^{m}(R_{a,i} - R_{b,i})^2}}$$

   b. The three similarity functions are defined where Ra,i; Rb,i; Ra, and Rb denote the ratings of users a and b on item i, average of all Ra,i and average of all Rb,i, respectively.

iii. Calculate CFPP

a. The rating information of the top k neighbors is then used to predict CF-based predicted preference of user a on item i, CFPP(a,i),

$$CFPP(a,i) = \overline{R}_a + \frac{1}{\sum_{b=1}^{k} |sim(a,b)|} \times \sum_{b=1}^{k} sim(a,b) \times (R_{b,i} - \overline{R}_b),$$

i. where k denotes the number of user a’s neighbors and sim(a,b) denotes the similarity between users a and b, which is computed using Pearson correlation coefficient, cosine similarity or distance measure.

iv. ***SPA***: Deriving sequential patterns

a. In order to calculating predicted preferences of items based on SPA method, sequence data of each user is generated firstly by sorting transaction data for the person according to the transaction date. Sequence data is a series of item sets, ordered by their purchase time stamp. And then, sequential patterns are derived from sequence data of users except a target user using the SPA method.

v. Calculate SPAPP

a. The support of matched sequential patterns, the SPA- based predicted preference of user a for candidate item i to recommend, SPAPP(a,i), is calculated using the following equation. $SPAPP(a,i) = \sum_{s \in SUB} Support_s^i$

i. where SUB denotes the set of all subsequences of user a, and $Support^i_s$ denotes the support of item i from a subsequence s.

vi. Integrate CFPP and SPAPP

a. CFPP and SPAPP are normalized to get N_CFPP and N_SPAPP, respectively, since they are different in the range of values. User a’s final predicted preference on item i, FPP(a,i), is calculated using the following equation:

$$FPP(a,i) = \alpha \times N\_CFPP(a,i) + (1 - \alpha) \times N\_SPAPP(a,i)$$

i. where a and 1 a are weights given to CF technique and SPA method, respectively, and a ranges from 0.0 to 1.0

Example:

Table 1
An example of implicit rating data derived from an original transaction data.

	Item1		Item2		Item3		Item4		Item5	
	Date	Rating	Date	Rating	Date	Rating	Date	Rating	Date	Rating
User1	01/01	3	–	–	01/02	1	01/03	5	–	–
User2	01/01	4	–	–	01/02	3	01/03	1	01/04	2
User3	–	–	01/01	1	01/02	2	–	–	01/03	4
User4	01/01	5	01/02	4	01/03	3	–	–	–	–
UserT	–	–	01/01	4	01/02	3	01/03	2	–	–

HOPE system first calculates CFPP values of all items by the target user T as follows. Neighbors of the user T can be identified by CF technique since the implicit rating information became available. To select the best similarity measure for this data, we have to calculate the similarity between a target user T and every other user using each of the three similarity measures. Cosine similarity is, however, assumed to be selected as the best similarity measure, which is thought to be sufficient for the purpose of illustration. From Table 1, the similarities between target user T and each one of user1, user2, user3, and user4 are 0.7071, 0.9648, 0.8944, and 1, respectively. Thus, if the number of neighbors is set to 2, the neighbors of target user T are users 2 and 4. Now, CFPP(T,1), CFPP(T, 2), CFPP(T, 3), CFPP(T, 4) and CFPP(T, 5) calculated using are 4.7455, 3.5, 3.2365, 2, and 3 as in table 2.

Table 2
The integration of results from CF-based and SPA-based recommendation system.

Predicted Items	CFPP	SPAPP	N_CFPP	N_SPAPP	FPP	Rank
Item1	4.7455	0	1	0	0.5	2
Item2	3.5	0	0.5463	0	0.2732	5
Item3	3.2365	1.25	0.4504	0.8333	0.6419	1
Item4	2	1.5	0	1	0.5	2
Item5	3	0.5	0.3642	0.3333	0.3488	4

HOPE system then calculates SPAPP values of all items by the target user T. The sequence data are constructed from Table 1 for the users except the target user T1:

User1: ⟨Item1⟩⟨Item3⟩⟨Item4⟩
User2: ⟨Item1⟩⟨Item3⟩⟨Item4⟩⟨Item5⟩
User3: ⟨Item2⟩⟨Item3⟩⟨Item5⟩
User4: ⟨Item1⟩⟨Item2⟩⟨Item3⟩

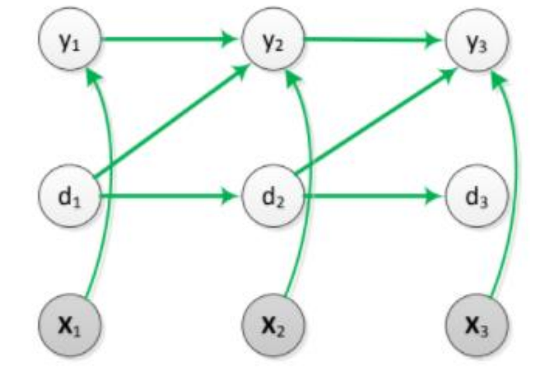
Suppose the minimum support 0.5 is given for sequential pattern mining. Then, the following list of sequential patterns with supports in parentheses is generated from the above sequence database:

⟨Item1⟩⟨Item3⟩(0.75)
⟨Item2⟩⟨ Item3⟩(0.5)
⟨Item3⟩⟨Item4⟩(0.5)
⟨Item3⟩⟨Item5⟩(0.5)
⟨Item1⟩⟨Item4⟩(0.5)
⟨Item1⟩⟨Item3⟩⟨Item4⟩(0.5)

Assume that the target user T has the sequence data <Item1> <Item3><Item2> in test data. The test data belong to B * D in Fig. 3. Its subsequences are <Item1>, <Item3>, <Item2>, <Item1><Item3>, <Item1><Item2>, <Item3><Item2> and <Item1><Item3><Item2>. By matching each subsequence with the starting part of the sequential patterns, we can decide candidate items to recommend and their support. For example, since the first subsequence hItem1i appears in the starting part of the first and the fifth sequential patterns, Item3 and Item4 can be decided as candidate items to recommend with supports 0.75 and 0.5. Similarly from the second subsequence <Item3>, both Item4 and Item5 are identified as candidate items to recommend with the same support 0.5 from the third and the fourth sequential patterns, respectively. After all candidate items to recommend are identified in this way, SPAPP(T, 1), SPAPP(T, 2), SPAPP(T, 3), SPAPP(T, 4), and SPAPP(T, 5) are calculated. They are 0, 0, 1.25(=0.75 + 0.5), 1.5(=0.5 + 0.5 + 0.5), and 0.5. See Table 2. When the weights given to CF technique and SPA method are set to 0.1 and 0.9, after conducting, respectively, FPP(T,1), FPP(T,2), FPP(T,3), FPP(T,4) and FPP(T, 5) calculated are 0.5, 0.2732, 0.6419, 0.5, and 0.3488, as shown in Table 2. Thus, if the number of items to recommend, n, is 3, then Item3, Item1, and Item4, which have the highest FPP values are selected as items to recommend by the HOPE system.

4.3 Recommendations based on life-stage

The importance of life stages on consumer's purchasing behaviours for many years, for an example consumers go through various life stages, such as bachelor stage (young & single), newly married couples (young & no children), full nest (married couple with dependent children), empty nest (elder married couples with no children living together). Full nest stage further can be divided into few substages. Therefore, in an E-commerce system, there is a strong correlation between life cycle stages and consumer purchasing behaviour. For an example, a woman will buy maternal vitamins during the pregnancy stage, then buy a baby car when the baby is born. A mum will have different needs during different life stages.



X: the behavior sequence of a user. Behavior sequence X of a user is a sequence of user actions ordered by time. User actions in our system include clicking/collecting/buying an item, issuing a query, clicking navigating tips, etc.

T: the length of X.

X_t: the observed behavior sequence at time t.

d_t: the duration of a life stage at time t.

y_t: the life stage label at time t.

l_{min},l_{max}: the minimum and maximum lengths of life stage respectively which are defined by domain expert-s.

The maximum entropy semi markov model for segmenting and labelling consumer life stages based on the observed purchasing data over time

Input: label each user's behaviour sequences to tell a user's current life-stage

Output: generate recommendations based on the predicted life-stage based recommendations, which includes how to model and predict life-stages and how to give appropriate product recommendation correspondingly

<u>Algorithm:</u>

i. The probability of life stage yt at time t depends on the previous life stage yt−1 at time t−1,

ii. How long the user has been in the previous life stage (i.e. dt−1) the observed user behavior sequence

iii. Given an observed behavior sequence X, our goal is to find the best underlying life-stage sequence y1, ..., yk and the corresponding duration di, ..., dk:

$$\{y_1, ..., y_k, d_i, ..., d_k\} = \underset{k,d_t,y_t}{\operatorname{argmax}} \prod_{t=1}^{k} P(y_t|y_{t-1}, d_{t-1}, X_t) P(d_t|d_{t-1})$$
$$\text{s.t. } l_{min} \leq d_t \leq l_{max}$$

a. where Xt denotes the observed behavior sequence at time t. T denotes the length of X. lmin and lmax are the minimum and maximum lengths of a stage, which are defined by do- main experts. p(yt|yt−1, dt−1, Xt) is the probability of being in yt at time t given the previous stage yt−1, the previous stage duration dt−1 and the observed behavior sequence Xt. The optimal segmentations and labels can be found using dynamic programming.

iv. That is the joint probability P(p productj , a), where a is the baby's age. Let P(p productj) be the probability of the user purchasing the product j, and we calculate it for each user-product pair based on the existed recommendation algorithms. Let P(a|p productj) be the conditional probability of making the purchase at age a given that the user will purchase product j. Based on the chain rule, we have:

$$P(p\_product_j, a) \\ = P(a|p\_product_j)P(p\_product_j)$$

v. Given a baby's age a, we can use the above value to rank all candidate products. For a user u without the age information, we first estimate his/her age distribution pu(a) based on the Gaussian mixture model, $P(p\_product_j) \int p(a|p\_product_j)p_u(a)da$

   a. where p(a|p productj) is the probability of the age of a random user purchasing product j (i.e. purchasing age), which we assume follows a Gaussian distribution N (μj , σ^2_j).

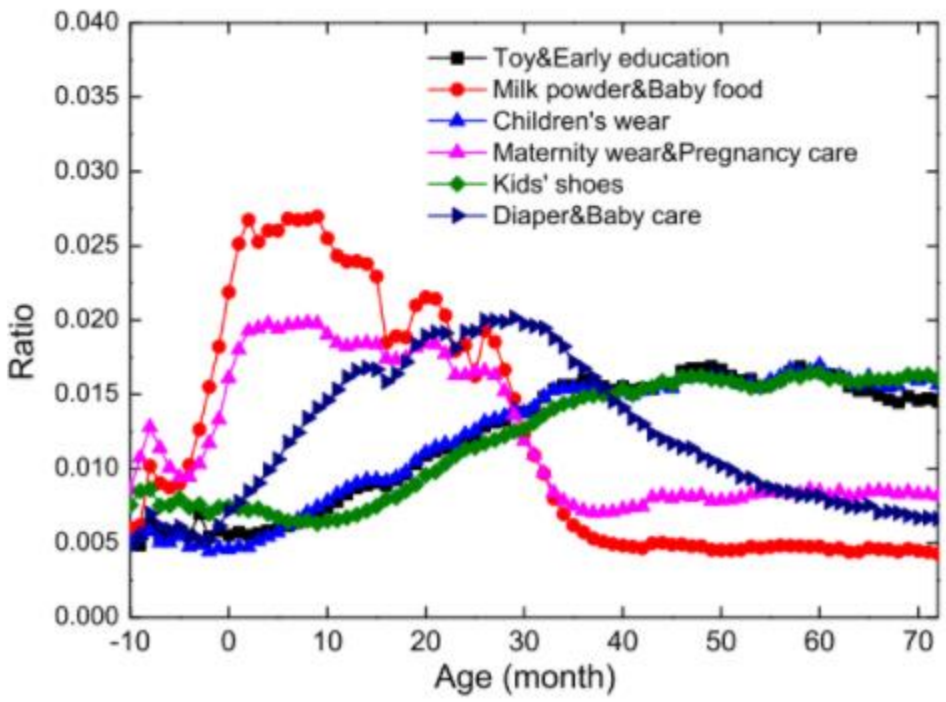
vi. To estimate P(p productj) in E.q 3, we use a large scale logistic regression model:

$$P(p\_product_j) = \frac{1}{1 + e^{-\mathbf{w}^T\mathbf{x}}}$$

   a. where x is a feature vector that represents a purchasing action, which includes scores generated by basic recommendation algorithms such as item-item based collaborative filtering, as well as other features that might be useful. w is learnt by maximizing the likelihood of the training data.

Example:

In the proposed idea, major models can be applied to various events, the experiments and large scale solution described in this paper are focusing on the vertical recommendation of mum-baby products. In the mum-baby vertical domain, the effect of life-stage on consumer behavior is the most obvious and very important. At different life stages of a baby, a woman needs to buy different matching products. For example, a woman is likely to buy maternity clothes at the prenatal stage, then she may look for milk powder after the baby is born. For a vertical recommender system, how to predict a user's life stage precisely and recommend appropriate products for the current life stage is a very important and challenging problem that will be solved by this approach.



The purchasing ratio of 6 top level categories changes with a baby's age (unit: month). We normalize the value of a category at each time stage using a category specific normalization factor (i.e. the total number of products in the category bought over all those years)

5. Conclusion

In this project report, Collaborative Filtering techniques are discussed in product recommendations region using sequential patterns, in E-commerce databases. Then trending algorithms for multiple E-commerce databases are discussed such as, RUM: Recommendation with User-memory network, Hybrid approach (this is not new, it is there in use for quite a long time, since last decade but its advantages are helpful in CF, so this one is popular as well), Life-stage prediction for product recommendation in E-commerce, etc. approaches were learnt and are discussed in this report to summarize the CF exploration to enhance product recommendations.

6. References

[1] D. Goldberg, D. Nichols, B. M. Oki and D. Terry, "Using collaborative filtering to weave an

information tapestry," 1992.

[2] X. Su and T. M. Khoshgoftaar, "A survey of Collaborative Filtering Techniques," in *Advances in Artificiall Intelligence*, 2009.

[3] P. Resnik and H. R. Varian, "Recommender systems," 1997.

[4] K. Goldberg, T. Roeder, D. Gupta and C. Perkins, "Eigentaste: a constant time collaborative filtering algorithm," in *Information Retrieval*, 2001.

[5] Wikipedia Contributors, "Collaborative Filtering," 07 March 2018. [Online]. Available: https://en.wikipedia.org/wiki/Collaborative_filtering.

[6] S. Rendle, C. Freudenthaler and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," in *WWW*, 2010.

[7] B. N. Miller, I. Albert, S. K. Lam, J. A. Konstan and J. Reidi, "MovieLens unplugged: experiences with an occasionally connected recommender system," in *International conference on intelligent user interfaces*, 2003.

[8] R. Mishra, P. Kumar and B. Bhaskar, "A web recommendation system considering sequential information," 2015.

[9] R. I. Mooney and C. Roy, "Content based book recommending using learning for textcategorization," in *ACM SIGIR Workshop recommender systems: algorithms and evaluation*, 1999.

[10] M. Pazzani and D. Bilsus, "Learning and revising user profiles: the identification of interesting websites," in *Machine Learning*, 1997.

[11] T. P. Liang, Y. F. Yang, D. N. Chen and Y. C. Ku, "A semantic expansion approach to personalized knowledge recommendation," in *Decision support systems*, 2004.

[12] X. Chen, H. Xu, Y. Zhang, J. Tang, Y. Cao, Z. Qin and H. Zha, "Sequential Recommendation with User Memory Networks," in *WSDM*, New York, NY, USA, 2018.

[13] D.-R. Liu, C.-H. Lai and W.-J. Lee, "A hybrid of sequential rules and collaborative filtering for product recommendation," in *Information Sciences*, 2009.

[14] K. Choi, D. Yoo, G. Kim and Y. Suh, "A hybrid online-product recommendation system:

combining implicit rating-based collaborative filtering and sequential pattern analysis," in *Electronic Commerce Research and Applications*, 2012.

[15] P. Jiang, Y. Zhu, Y. Zhang and Q. Yuan, "Life-stage prediction for product recommendation in E-commerce," in *KDD*, Sydney, NSW, Australia, 2015.

[16] Q. Zhao, "E-commerce product recommendation by personalized promotion and total surplus maximization," in *WSDM*, San Francisco, CA, USA, 2016.

[17] Q. Zhao, Y. Zhang, D. Friedman and F. Tan, "E-commerce recommendation with personalized promotion," in *RecSys*, Vienna,Austria, 2015.