Abstract—The existing Collaborative Filtering (CF) technique that has been widely applied by e-commerce sites requires a large amount of ratings data to make meaningful recommendations. It is not directly applicable for recommending products that are not frequently purchased by users, such as cars and houses, as it is difficult to collect rating data for such products from the users. Many of the e-commerce sites for infrequently purchased products are still using basic search-based techniques whereby the products that match with the attributes given in the target user’s query are retrieved and recommended to the user. However, search-based recommenders cannot provide personalized recommendations. For different users, the recommendations will be the same if they provide the same query regardless of any difference in their online navigation behaviour. This paper proposes to integrate collaborative filtering and search-based techniques to provide personalized recommendations for infrequently purchased products. Two different techniques are proposed, namely CFRRobin and CFAgQuery. Instead of using the target user’s query to search for products as normal search based systems do, the CFRRobin technique uses the products in which the target user’s neighbours have shown interest as queries to retrieve relevant products, and then recommends to the target user a list of products by merging and ranking the returned products using the Round Robin method. The CFAgQuery technique uses the products that the user’s neighbours have shown interest in to derive an aggregated query, which is then used to retrieve products to recommend to the target user. Experiments conducted on a real e-commerce dataset show that both the proposed techniques CFRRobin and CFAgQuery perform better than the standard Collaborative Filtering (CF) and the Basic Search (BS) approaches, which are widely applied by the current e-commerce applications. The CFRRobin and CFAgQuery approaches also outperform the existing query expansion (QE) technique that was proposed for recommending infrequently purchased products.

Keywords — recommender system; collaborative filtering; search-based technique; personalization

I. INTRODUCTION

The exponential growth of the World Wide Web (WWW) has changed how we conduct our daily activities.
current Collaborative Filtering (CF) approach is not directly applicable for recommending this kind of product.

Currently, the standard search engines are still widely applied as the common tool for users to search for the expensive and infrequently purchased products. In this kind of search, users are required to specify product attributes as a query and the search engine matches the query with a set of available products in the database to retrieve a list of products that will most likely be of interest to the user. Although the standard search engine is simple to implement, the search results are not personalized as only products that have the same attributes as the users’ queries will be displayed to them. For different users, the recommendations will be the same if they provide the same query no matter how different their online navigation behaviour is. In addition, the users may not know the technical details of the products that they want to buy, and, thus, very often they are not able to provide accurate or sufficient information in their query to the search engine.

In this paper, we propose to integrate the collaborative filtering and search-based techniques to generate personalized recommendations for infrequently purchased products. Instead of using the product attributes given by the target user in his or her query to retrieve products, we propose to apply a collaborative filtering technique to find similar users (called neighbours) who have similar behaviour or preferences to the target user; then, based on the products or the attributes of the products in which the neighbours have shown interest, product recommendations for the target user are generated. Two different methods are proposed in this paper. The first method – CFRRobin – integrates the CF technique with the Round Robin fusion technique [4]. It first generates multiple queries based on each of the products of the target user’s neighbours to retrieve relevant products, and then merges and ranks the retrieved products using the Round Robin method. The second method – CFAgQuery first derives more detailed product attributes based on the product preferences of the neighbours, and then generates a new query by aggregating the derived product attributes. The new query captures the user preferences of the neighbour and provides more detailed content that may be of interest but may have been missed by the target user when she/he submitted her/his query. Product recommendations will be generated based on the new query and also the similarity between the target user and the neighbours. By using the preferences of the target user’s neighbours, the queries, and, thus, the recommendations are personalized to the target user’s interests. The two methods will be discussed in Section III-B.

For the collaborative filtering technique, user preference information is essential. Usually, user previous rating data is used to profile a user’s item/product preferences. In the case of a lack of user rating data, users’ online search behaviours can be utilized to create users’ profiles, which are then used to identify similar users or neighbours of the target user. In this paper, we propose a method to generate users’ product interests/preferences based on users’ online navigation log data, which is introduced in Section III-A.

The paper is organized as follows. First, the related work will be briefly reviewed in Section II. Then, the proposed methods will be explained in detail in Section III. Next, Section IV provides the experiments and evaluation results. Finally, the conclusion will be given in Section V.

II. RELATED WORK

There are two kinds of product offered to users on e-commerce sites – low involvement products (LIP), such as books, videos, soap, and high involvement products (HIP), such as electronic devices, cars, houses [4]. The Collaborative Filtering (CF) approach has been widely applied for recommending LIP because a rich source of data is available for learning users’ preferences and for generating personalized recommendations according to the preference of similar users. The CF approach works best with a large amount of user preferences data and it is suitable for recommending LIP that are frequently purchased by users, as its database of users’ preferences gets larger and larger over time when users purchase the products repetitively. Thus, currently, the CF approach has been widely employed by many of the commercial retail websites for recommending LIP.

The CF approach is not directly applicable for recommending HIP as the products are not frequently purchased by the users during their lifetime, and users are not able to provide ratings for products they never use. Currently, many of the e-commerce websites are still implementing the standard search-based approach for recommending HIP in which the user has to specify product attributes as the query, and the user’s input is matched with the available products in the database to retrieve products that will most likely be of interest to the user. However, the user’s initial query is normally short and does not fully represent the user’s requirements. The query expansion approach has been proposed in [5] to expand the user’s query based on the associations between the product attribute values extracted from products that have received positive reviews from the previous users. In the literature, recommendations for HIP are also resolved as a product selection problem by using approaches like Case-Based Reasoning and multi-criteria decision analysis. However, in recommending products none of these methods provide personalized recommendations, as they do not predict the users’ preferences for use in product recommendations.

The current approaches for recommending HIP requires high involvement from users to provide product attributes that are of interest to them as queries. While, the CF approach, which is widely applied for LIP requires sufficient ratings or purchases history data to generate meaningful recommendations. Methods that can learn users’ profiles without the availability of user’s ratings or requires
high involvement from the users are needed for providing personalized recommendations for the HIP.

The usage of implicit feedback for recommending products has attracted new developments in recommendation algorithms that are suitable for processing implicit feedback. [4] proposed a recommendation methodology for HIP based on the users’ profiles, which are generated using the user’s past purchases. Their method utilizes the specified user’s multi-attributes and preferences from data from past purchases for recommending products using the CF approach. Their method assumes that the user has purchased a set of products in the related product category in the past. [6] proposed to transform the implicit user observations into two paired magnitudes, namely, preference and confidence levels. Confidence scores are determined from the frequency of actions, such as the frequency that a user bought a certain item. These confidence scores are attached to the estimated preferences to indicate whether the user’s preference is positive or negative. They proposed a latent factor algorithm that addresses the preference-confidence paradigm to tailor for implicit feedback recommendations. [7] incorporated temporal information, such as user purchase time and item launch time, to construct pseudo rating data from the user purchase information for collaborative filtering. Instead of simply assigning 1 to the purchased items, a rating function is defined that computes rating values based on the launch time and purchase time of items to reflect the users’ preferences to achieve better recommendation accuracy. [8] proposed a Collaborative Filtering based recommender system that utilizes the preference levels of a user for a product, which are estimated from the navigational and behavioural patterns of users. The preference level of a purchased product is set to 1 and the preference level of a product that is clicked but not purchased is estimated based on the probability of a product being purchased, which is calculated based on certain variables captured in the navigational data such as number of visits, length of reading time, and basket placement status.

The recommendation algorithms for processing implicit feedback are often studied independently from the domain knowledge. For HIP, the product features are an important factor for the user to consider when making a decision about the final products to buy. The current generation of recommender systems require further improvements to make recommendation methods more effective and applicable to an even broader range of real life applications, which includes recommendations pertaining to more complex types of application [9]. This paper proposes to incorporate knowledge about product attribute values to generate user profiles from user click stream data. The profiles are then utilized by the CF approach to find similar users or neighbours who have the same preferences as the target user. The search-based approach is also integrated to search for products having attribute values similar to those products that have been of interest to the target user’s neighbours.

III. PROPOSED APPROACHES

This section first defines certain concepts and entities used in this paper, and then introduces the proposed methods to generate users’ profiles and the methods to recommend users with the most relevant products by combining the collaborative filtering technique and search based techniques.

- **Product**: Product refers to any type of product or online services for which users can search for information or purchase. Usually a product can be described by a set of attributes and each attribute can have a set of possible values. Suppose that there are \( n \) attributes \( A_1, A_2, ..., A_n \) for a product \( P \), each attribute \( A_i \) has a set of possible values, \( \{a_{i1}, a_{i2}, ..., a_{im_i}\} \), a product \( P \) can be represented by a vector of attribute values, i.e., \( P = <a_1, a_2, ..., a_n> \) and \( a_i \in \{a_{i1}, a_{i2}, ..., a_{im_i}\} \), \( i = 1, 2, ..., n \). For example, attributes for car domain may include “make”, “model”, “year”, “body type”, “price” and “transmission”. If attribute \( A_i \) is “body type”, the possible attribute values for \( a_i \) can be “coupe”, “hatchback”, “sedan” and “wagon”.

- **User Session**: A user session \( S \) represents a user’s online click stream that contains a series of products viewed by the user. Let \( S \) be a set of products viewed by a user, i.e., \( S = \{p^1, p^2, ..., p^{|S|}\} \), each product can be represented as a vector of attribute values: \( p^k = <a^k_1, a^k_2, ..., a^k_{|S|}> , k = 1, 2, ..., |S| \) and \( a^k_i \in \{a_{i1}, a_{i2}, ..., a_{im_i}\} \). Each product can also be represented as a set of attribute values: \( p^k = \{A_1 = a^k_1, A_2 = a^k_2, ..., A_n = a^k_n\} \).

A. User Profiling

User profiling aims to generate a user profile that represents the user’s preference or interest in products or product attributes. When a user searches for products to buy, the user will usually click on some products to view. The user’s product clicks show that the user has more interest in the viewed products compared to the other products. This online click stream data contains valuable information that can be used to predict the user’s interests or preferences to the products. In the method to be described in this section, this data is used to generate user profiles. A user profile represents a set of user’s preferences to each attribute value of a product, which shows how much interest the user has in the attribute of the products.

From a set of products viewed by a user, i.e., \( S = \{p^1, p^2, ..., p^{|S|}\} \), if each product is treated as a transaction of attribute values, i.e., \( \{A_1 = a_1, A_2 = a_2, ..., A_n = a_n\} \) for a
product \( P = \langle a_1, a_2, \ldots, a_n \rangle \), from \( S \), a product transaction dataset of \(|S|\) transactions can be constructed for the user. From the transaction dataset, the frequency \( \text{freq}(a_{ij}) \) of each attribute value \( a_{ij} \) for attribute \( A_i \) can be obtained. In this paper, we propose to represent a user’s product interest by using the user’s preferences to product attribute values. A user profile is represented as \( \text{up} = \langle u_{a_{11}}, \ldots, u_{a_{1m}}, u_{a_{21}}, \ldots, u_{a_{2m}}, \ldots, u_{a_{nm}} \rangle \), where \( u_{a_{ij}} \) denotes the user’s preference to the \( j \)th value of attribute \( A_i \). The user’s profile can be calculated based on the frequency of \( a_{ij} \) obtained from the user’s product transaction dataset, as follows:

\[
u_{a_{ij}} = \frac{\text{freq}(a_{ij})}{|S|}
\]

Since \( \sum_{j=1}^{m} \text{freq}(a_{ij}) = |S| \), \( \sum_{j=1}^{m} u_{a_{ij}} = 1 \).

B. Product Retrieval

Based on user profiles, we can calculate users’ similarity in terms of their item/product preferences. Let \( T = \langle t_1, t_2, \ldots, t_m \rangle \) be a target user’s profile, where \( m \) is the total number of attribute values, and \( V_i = \langle v_{i1}, v_{i2}, \ldots, v_{im} \rangle \) be a previous user’s profile. The neighbourhood formation aims to calculate the preference similarities between the target user profile \( T \) and the previous user \( V_i \). The similarity value between the two users can be calculated by using a similarity method, such as the cosine similarity given below:

\[
sim(T, V_i) = \frac{\sum_{j=1}^{m} t_{ij} \cdot v_{ij}}{\sqrt{\sum_{j=1}^{m} t_{ij}^2} \cdot \sqrt{\sum_{j=1}^{m} v_{ij}^2}}
\]

The top-\( R \) previous users who are very similar to the target user are selected as the target user’s neighbours. For the standard collaborative filtering technique, the products that are preferred by the neighbours will be used as the candidates to generate the recommendations. We could use the standard CF method to recommend products that are preferred by neighbours. This is actually one of the baseline methods that are used in our experiments to evaluate the performance of the proposed methods CFRRobin and CFAGQuery. For online infrequent purchased product searches, there is a problem for directly using the standard CF method. For expensive products, such as houses or used cars, the products that previous users have purchased or viewed may no longer be available. Therefore, directly recommending products purchased or viewed by previous users becomes meaningless since those products may not exist anymore. For solving this problem, in this paper, we propose to recommend products that have similar attributes to the products preferred by the user’s neighbours rather than directly recommending the user’s neighbours’ preferred products. Below we introduce two proposed methods, CFRRobin and CFAGQuery, to generate a set of candidate products, and a ranking method to select the most relevant products to recommend.

1) CFRRobin

This method incorporates the fusion technique Round Robin method [10] to the CF technique to generate a set of candidate products. By using the CF technique, we can generate a set of neighbours for the target user. Let \( \{B_1, B_2, \ldots, B_\beta\} \) be the set of neighbours of the target user, \( S_{B_i} = \{p_{i1}, p_{i2}, \ldots, p_{i|S_{B_i}|}\} \) represents a set of products viewed by a neighbour \( B_i \). Instead of using the products in \( S_{B_i} \) as the candidates for recommendations, the attributes of each of the products are used as a query to retrieve products from the database that have similar attributes. That is, \( \forall p^{ij} \in S_{B_i}, Q^{ij} = \{A_1 = a_{ij}, A_2 = a_{ij}, \ldots, A_n = a_{ij}\} \) is a query containing the attributes of a product that the neighbour \( B_i \) is interested in. A set of products, \( \{b_1, b_2, \ldots\} \), whose attributes match the attributes in \( Q^{ij} \) can be retrieved and also ranked based on the similarity \( \text{sim}(b_k, Q^{ij}) \) between the products \( b_k \) and the query \( Q^{ij} \). Generally, the attribute values are not necessarily numerical values, they can be nominal attributes. For numerical attributes, the cosine similarity can be used to measure the similarity. For nominal attributes, let \( b_k = \{a_1 = a_{ij}, a_2 = a_{ij}, \ldots, a_n = a_{ij}\} \), the following method can be used to measure the similarity:

\[
\text{sim}(b_k, Q^{ij}) = \sum_{i=1}^{n} \text{sim}_A(a_{ij}, a_{ij}) = \begin{cases} 1, & a_i = a_{ij} \\ 0, & a_i \neq a_{ij} \end{cases}
\]

\( \forall p^{ij} \in S_{B_i}, \) based on the similarity, a list of ranked products can be generated, \( L^{ij} = \langle b_1^{ij}, b_2^{ij}, \ldots, b_\beta^{ij} \rangle \). Therefore, from the neighbour \( B_i \), \( |S_{B_i}| \) lists of products are generated: \( L^{i1}, L^{i2}, \ldots, L^{i|S_{B_i}|} \). All the products in these lists are similar to the products preferred by \( B_i \) in terms of the product attributes. By applying the Round Robin method to the lists, we can rank all the products in \( L^{i1} \cup \ldots \cup L^{i|S_{B_i}|} \). The Round Robin method selects a product from the top of each \( L^{ij} \) for each round, and then starts again from the top of the list for the remaining products in each \( L^{ij} \). From the ranked products in \( L^{i1} \cup \ldots \cup L^{i|S_{B_i}|} \), the top \( N \) products are chosen as the candidates generated from neighbour \( B_i \), denoted as \( C_{B_i} \). Thus, by combining the products in \( C_{B_i} \) for all neighbours, we obtain a set of candidate products.

\[
C = \bigcup_{i=1}^{\beta} C_{B_i}
\]

2) CFAGQuery

In this approach, instead of using each product as a query, an aggregated query is first derived based on the
products viewed by all the user’s neighbours and then the aggregated query is used to retrieve relevant products. Let \( \{B_1, B_2, ..., B_R\} \) be a set of user’s neighbours, \( S_{B_i} = \{P^{(1)}_i, P^{(2)}_i, ..., P^{(|S_{B_i}|)}_i\} \) represents a set of products viewed by the user’s neighbour \( B_i \). Using the profiling method introduced in Section III-A, from \( S_{B_i} \), a product transaction dataset of \( |S_{B_i}| \) transactions can be constructed for \( B_i \). From the transaction dataset, the frequency \( f_{req}(a_{ij}) \) of each attribute value \( a_{ij} \) for attribute \( A_j \) can be obtained. The same as for profiling a user, the neighbour \( B_i \)’s product interest, denoted as:

\[
up_i = < u_{i1}, ..., u_{i1m_1}, u_{i2}, ..., u_{i2m_2}, ..., u_{im_n} >
\]

where \( u_{ij} \) denotes the user’s preference to the \( j \)th value of attribute \( A_k \), can be calculated as follows:

\[
u_{ij} = \frac{f_{req}(a_{ij})}{|S_{B_i}|} \quad \text{(4)}
\]

By combining \( u_{ij}^1, u_{ij}^2, ..., u_{ij}^\# \), we can generate an aggregated profile

\[
u_{ij}^ag = < u_{i1}^{ag}, ..., u_{i1m_1}^{ag}, u_{i2}^{ag}, ..., u_{i2m_2}^{ag}, ..., u_{im_n}^{ag} >
\]

for the target user \( u \). Based on the preferences of the target user’s neighbours, the preference of the target user \( u \) to each attribute value \( u_{kj}^{ag} \) of attribute \( A_k \) can be calculated using the following equation:

\[
u_{kj}^{ag} = \frac{\sum_{i=1}^\#(sim(u, B_i) \times u_{kj})}{\sum_{i=1}^\#sim(u, B_i)} \quad \text{(5)}
\]

where \( sim(u, B_i) \) is the similarity between \( u \) and its neighbour \( B_i \) which can be calculated using the cosine similarity.

\( u_{kj}^{ag}, ..., u_{kj}^{ag} \) measure the preference strength of the target user to each attribute value of attribute \( A_k \) based on the viewpoint of the target user’s neighbours. It is easy to prove that \( \sum_{k=1}^{\#} u_{kj}^{ag} = 1 \). By choosing the attribute value with the highest preference for each attribute, we can generate an aggregated query \( AQ^u = \{A_1 = a_1^{ag}, A_2 = a_2^{ag}, ..., A_n = a_n^{ag}\} \)

where \( a_k^{ag} = \max_{j=1}^{m_k} (u_{kj}^{ag}) \). Then, by doing a search of the product database, products that match the aggregated query \( AQ^u \) are retrieved as candidate products \( \Gamma \) for the target user.

In the next section, we discuss a ranking method to rank the candidate products generated by CFRRobin or CFAgQuery. The top N products will be recommended to the target user.

C. Product Ranking

The final process is to rank the products in the candidate list and to select the top-N products to recommend. The products are ranked based on the similarities between each product and the target user’s interests.

Let the target user’s profile be

\[
up^u = < u_{1}, ..., u_{1m_1}, u_{2}, ..., u_{2m_2}, ..., u_{nm_n} >
\]

which is generated from the target user’s online click data, by choosing the attribute value with the highest preference for each attribute, we can generate the target user’s preferred attribute values \( Q^u = \{A_1 = a_1^u, A_2 = a_2^u, ..., A_n = a_n^u\} \)

where \( a_k^u = \max_{j=1}^{m_k} (u_{kj}^u) \). Let \( \Gamma \) be the set of candidate products generated by CFRRobin or CFAgQuery, \( \beta_k \in \Gamma \) and \( b_k = \{A_1 = a_1, A_2 = a_2, ..., A_n = a_n\} \), the similarity between \( b_k \) and \( Q^u \), denoted as \( sim(b_k, Q^u) \), is used to rank the products in \( \Gamma \). The similarity \( sim(b_k, Q^u) \) can be calculated using Equation (3). Finally, the top-N products are selected as the final products to recommend from the ranked products in the candidate list.

IV. EXPERIMENT AND EVALUATION

A. Datasets

A case study was conducted for the car online selling domain. Data were collected from one of the car selling websites. The dataset contained 17,690 cars and 20,868 user navigation sessions generated from the web search log. Only sessions with at least four viewed cars were selected for the experiments. The final dataset contained 3,564 user sessions in which each session represented a sequence of cars that had been viewed by a user. Three other search models were developed as the baseline models— the Basic Search (BS), Query Expansion (QE) proposed in [5] and Original Collaborative Filtering (CFOriginal). The BS model is the basic search technique, which retrieves cars that have attribute values that match the user’s query terms. Usually, only basic ranking is implemented by online search engines, that is, the retrieved products are ranked by one of the product’s attributes, for instance, price or car model. In this experiment, the retrieved cars were ranked by comparing the similarity between the attributes of the retrieved cars and the attributes of the query used for that retrieval. The QE model extends the target user’s query to other attribute values based on users’ online product reviews. The CFOriginal model implements basic collaborative filtering, which directly recommends cars that have been viewed by the target user’s neighbours. The proposed methods – CFRRobin and CFAgQuery were evaluated in the experiment against the three baseline models.

The user session dataset was partitioned into 5 sub datasets. Each of them (20% of user sessions) was used as a testing dataset and the remaining part was used as training data. Each session in the testing dataset was further divided into two parts evenly. As a result, the session dataset contains three parts – Training, Testing Part 1 and Testing Part 2, as illustrated in Figure 1.
Training and Testing Part 1 were used to generate previous users’ profiles and target users’ profiles, respectively. The profiles were used for neighbourhood formation for the CFOriginal, CFRRobin and CFAgQuery models. The sessions in Testing Part 1 were considered as target users and the cars listed in each session were considered as cars viewed by the target users. Sessions in the training data were considered as previous users, which were used to find the target user’s neighbours. For each neighbour, the cars in the training dataset were considered as the cars in which the previous user was interested. Moreover, the last car for each session in Testing Part 1 was also used as the query to retrieve cars for the BS and QE. The recommendations generated by the BS, QE, CFOriginal, CFRRobin and CFAgQuery models for each session in Testing Part 1 were matched with the cars in the same session. For each experiment, there were 5 runs. Finally, the average result for the 5 runs was calculated.

In this experiment, we tested whether the proposed models - CFRRobin and CFAgQuery outperform the baseline models - BS, QE, and CFOriginal, and the impact of using different user profiles created by using different amounts of the target user’s click data. In order to appropriately achieve the second purpose, five sets of user profiles were generated for the users in the testing dataset using the proposed method discussed in Section III-A, UT1, UT2, UT3, UT4, UT5, which contain user profiles generated using the last viewed car, the last 2 viewed cars, the last 3 viewed cars, the last 4 viewed cars, and the last 5 viewed cars by the target user, respectively.

We designed five runs of experiments for CFOriginal, CFRRobin and CFAgQuery models, as shown in Table 1. For the BS and the QE models, which do not implement the collaborative filtering technique, only the last car is used as the query to retrieve relevant cars. Table 1 shows different runs of experiments for the BS, QE, CFOriginal, CFRRobin and CFAgQuery models.

**B. Evaluation Metrics**

In this evaluation, we do not match the retrieved car ID exactly with the car IDs in the testing dataset because for the car domain, in which we conduct this experiment, different car IDs may refer to different cars that have the same attributes. The purpose of product searching is to provide users with the products that meet users’ requirements on product attributes or features. In this experiment, to evaluate all the models, for each session, if at least 80% of the attributes of a retrieved car match the attributes of one of the cars in the same session of the Testing Part 2, the retrieved car was considered as matching the testing car. The focus of this experiment is to recommend cars that match the attribute values preferred by the user. Thus, we may recommend cars with different IDs but that have the same attributes.

In this experiment, we evaluate the performance of the five models by retrieving the top 5, 10, 15, 20, 25 and 30 most relevant cars. Two evaluation metrics were used in this experiment – recall and precision:

\[
\text{Recall} = \frac{NM}{NT}
\]

\[
\text{Precision} = \frac{NM}{NR}
\]

where \(NM\) is the number of retrieved cars that match with the testing cars, \(NT\) is the number of testing cars in the session, and \(NR\) is the number of retrieved cars. Finally, the average recall and precision for all sessions (i.e., all users) were calculated for each search model.

**TABLE 1. DIFFERENT RUNS OF THE EXPERIMENTS**

<table>
<thead>
<tr>
<th>Models</th>
<th>Runs</th>
<th>User Profiles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Search (BS)</td>
<td>BS</td>
<td>UT1</td>
</tr>
<tr>
<td>Query Expansion (QE)</td>
<td>QE</td>
<td>UT2</td>
</tr>
<tr>
<td>Original Collaborative Filtering (CFOriginal)</td>
<td>CFOriginal1Cars</td>
<td>UT3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CFOriginal2Cars</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CFOriginal3Cars</td>
</tr>
<tr>
<td>Collaborative Filtering with Round Robin (CFRRobin)</td>
<td>CFRRobin1Cars</td>
<td>UT6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CFRRobin2Cars</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CFRRobin3Cars</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CFRRobin4Cars</td>
</tr>
<tr>
<td>Collaborative Filtering with query aggregation (CFAgQuery)</td>
<td>CFAgQuery1Cars</td>
<td>UT10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CFAgQuery2Cars</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CFAgQuery3Cars</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CFAgQuery4Cars</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CFAgQuery5Cars</td>
</tr>
</tbody>
</table>

**C. Results**

The precision results are given in Table 2 to Table 6 and the recall results are given in Table 7 to Table 11 for different target users’ profiles.

The precision results show that our proposed approaches, CFRRobin and CFAgQuery, perform better than all the three baseline models BS, QE and CFOriginal. The precision results for the CFAgQuery and CFRRobin models are quite similar for the profile generated from the last car. For the profiles generated from more cars, the
precisions for the CFAgQuery model are better than the precisions for the CFRRobin model.

The recall results show that, on average, the CFRRobin and CFAgQuery models outperform the BS, QE, and the CFSOriginal models. For profiles created from the last 2 and 3 cars, both the CFRRobin and CFAgQuery models perform better than all three baseline models, BS, QE, and the standard CF. However, for profiles created from the last one car (Table 7), the last 4 cars, or the last 5 cars, the recall results for the CFRRobin and CFAgQuery models are not always better than the BS model, but still better than the QE and the CFSOriginal models.

In summary, the results show that in terms of precision, for all user profiles, the CFAgQuery and CFRRobin models perform better than all three baseline models, i.e., BS, QE, and the CFSOriginal models. In terms of recall, for profiles generated from the last 2 cars or 3 cars, the CFAgQuery and CFRRobin models always perform better than all three baseline models; for profiles generated from the last one car, or the last 4 or 5 cars, the CFAgQuery and CFRRobin models perform better than the QE and the CFSOriginal models, but not always better than the BS model.

By using only one car to create the target user profiles, the profiles become very focused and the proposed models may recommend many similar cars. If a few of the last viewed cars are considered to create user profiles, i.e., the last 2 or 3 cars, the recall results of the CFAgQuery and CFRRobin are improved because the profiles become more diverse and the proposed methods can retrieve more cars that satisfy the users’ needs. However, when more cars are considered, i.e., the last 4 or 5 cars, the recall results of the CFAgQuery and CFRRobin models are not better than the BS model. This shows that only a few recent viewed cars, i.e., last 2 or 3 cars should be considered to generate the target user’s profile. The results for the CFAgQuery and CFRRobin models are good for both precision and recall for these profiles because the user has more interest in the cars that have been recently viewed.

### TABLE 2: PRECISION RESULTS OF THE BS, QE, CFSORIGINAL1CARS, CFRROBIN1CARS, CFAGQUERY1CARS

<table>
<thead>
<tr>
<th></th>
<th>Top 5</th>
<th>Top 10</th>
<th>Top 15</th>
<th>Top 20</th>
<th>Top 25</th>
<th>Top 30</th>
</tr>
</thead>
<tbody>
<tr>
<td>BS</td>
<td>0.41</td>
<td>0.43</td>
<td>0.43</td>
<td>0.42</td>
<td>0.42</td>
<td>0.42</td>
</tr>
<tr>
<td>QE</td>
<td>0.49</td>
<td>0.49</td>
<td>0.48</td>
<td>0.47</td>
<td>0.47</td>
<td>0.46</td>
</tr>
<tr>
<td>CFSOriginal1Cars</td>
<td>0.08</td>
<td>0.07</td>
<td>0.07</td>
<td>0.06</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>CFRRobin1Cars</td>
<td>0.64</td>
<td>0.64</td>
<td>0.62</td>
<td>0.61</td>
<td>0.60</td>
<td>0.59</td>
</tr>
<tr>
<td>CFAGQuery1Cars</td>
<td>0.64</td>
<td>0.63</td>
<td>0.62</td>
<td>0.61</td>
<td>0.60</td>
<td>0.59</td>
</tr>
</tbody>
</table>

### TABLE 3: PRECISION RESULTS OF THE BS, QE, CFSORIGINAL2CARS, CFRROBIN2CARS, CFAGQUERY2CARS

<table>
<thead>
<tr>
<th></th>
<th>Top 5</th>
<th>Top 10</th>
<th>Top 15</th>
<th>Top 20</th>
<th>Top 25</th>
<th>Top 30</th>
</tr>
</thead>
<tbody>
<tr>
<td>BS</td>
<td>0.41</td>
<td>0.43</td>
<td>0.43</td>
<td>0.42</td>
<td>0.42</td>
<td>0.42</td>
</tr>
<tr>
<td>QE</td>
<td>0.49</td>
<td>0.49</td>
<td>0.48</td>
<td>0.47</td>
<td>0.47</td>
<td>0.46</td>
</tr>
<tr>
<td>CFSOriginal2Cars</td>
<td>0.08</td>
<td>0.06</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.04</td>
</tr>
<tr>
<td>CFRRobin2Cars</td>
<td>0.60</td>
<td>0.60</td>
<td>0.60</td>
<td>0.58</td>
<td>0.57</td>
<td>0.56</td>
</tr>
<tr>
<td>CFAGQuery2Cars</td>
<td>0.61</td>
<td>0.61</td>
<td>0.60</td>
<td>0.59</td>
<td>0.59</td>
<td>0.58</td>
</tr>
</tbody>
</table>
TABLE 10. RECALL RESULTS OF THE BS, QE, CFORIGINAL4CARS, CFRRobin4Cars, CFAGQuery4Cars

<table>
<thead>
<tr>
<th></th>
<th>Top 5</th>
<th>Top 10</th>
<th>Top 15</th>
<th>Top 20</th>
<th>Top 25</th>
<th>Top 30</th>
</tr>
</thead>
<tbody>
<tr>
<td>BS</td>
<td>0.27</td>
<td>0.30</td>
<td>0.30</td>
<td>0.30</td>
<td>0.30</td>
<td>0.30</td>
</tr>
<tr>
<td>QE</td>
<td>0.23</td>
<td>0.23</td>
<td>0.24</td>
<td>0.25</td>
<td>0.26</td>
<td>0.27</td>
</tr>
<tr>
<td>CFROriginal4Cars</td>
<td>0.12</td>
<td>0.12</td>
<td>0.13</td>
<td>0.14</td>
<td>0.14</td>
<td>0.16</td>
</tr>
<tr>
<td>CFRRobin4Cars</td>
<td>0.27</td>
<td>0.30</td>
<td>0.30</td>
<td>0.33</td>
<td>0.34</td>
<td>0.36</td>
</tr>
<tr>
<td>CFAGQuery4Cars</td>
<td>0.28</td>
<td>0.29</td>
<td>0.29</td>
<td>0.29</td>
<td>0.31</td>
<td>0.31</td>
</tr>
</tbody>
</table>

TABLE 11. RECALL RESULTS OF THE BS, QE, CFORIGINAL5CARS, CFRRobin5Cars, CFAGQuery5Cars

<table>
<thead>
<tr>
<th></th>
<th>Top 5</th>
<th>Top 10</th>
<th>Top 15</th>
<th>Top 20</th>
<th>Top 25</th>
<th>Top 30</th>
</tr>
</thead>
<tbody>
<tr>
<td>BS</td>
<td>0.27</td>
<td>0.30</td>
<td>0.30</td>
<td>0.30</td>
<td>0.30</td>
<td>0.30</td>
</tr>
<tr>
<td>QE</td>
<td>0.23</td>
<td>0.23</td>
<td>0.24</td>
<td>0.25</td>
<td>0.26</td>
<td>0.27</td>
</tr>
<tr>
<td>CFROriginal5Cars</td>
<td>0.08</td>
<td>0.10</td>
<td>0.11</td>
<td>0.11</td>
<td>0.11</td>
<td>0.12</td>
</tr>
<tr>
<td>CFRRobin5Cars</td>
<td>0.27</td>
<td>0.28</td>
<td>0.29</td>
<td>0.33</td>
<td>0.34</td>
<td>0.35</td>
</tr>
<tr>
<td>CFAGQuery5Cars</td>
<td>0.26</td>
<td>0.26</td>
<td>0.27</td>
<td>0.28</td>
<td>0.29</td>
<td>0.29</td>
</tr>
</tbody>
</table>

I. CONCLUSION

In this paper, we investigated the methods for recommending infrequently purchased products by integrating collaborative filtering techniques and search-based techniques. We utilize users’ online click stream data to learn users’ preferences for creating users’ profiles. Two methods are proposed, which are the CFRRobin and CFAGQuery. Both methods generate the target user’s product preferences (i.e., the target user profile) based on the target user’s neighbours’ preferences using a frequent-based technique. Instead of directly recommending the products that the user’s neighbours have liked as implemented in the standard collaborative filtering, the proposed methods search the product dataset by using the generated target user’s profile as a query for products which match the target user’s preferences.

The experiment results show the proposed methods perform better than the Basic Search (BS), the Query Expansion (QE), and the Original Collaborative Filtering (CFOriginal) models. We compare the performances of different target user profiles. The result shows that only limited number of the previous products should be considered for creating users’ profiles. This is because only the most recently viewed products represent the products that the user is really interested in and the profile created using these most recently viewed products can represent the user’s preferences more accurately. For the future work, instead of using frequent-based technique to create users’ profiles, we intend to apply model-based techniques such as probabilistic latent semantic analysis to learn the user’s preferences and to create the users’ profiles in order to improve the recommendations generated by the proposed methods.

REFERENCES