# Exploiting the Interplay among Products for Efficient Recommendations

Anbarasu Sekar[1], Sutanu Chakraborti[2]
Dept. of Computer Science and Engineering
Indian Institute of Technology Madras
Chennai, India 600036

***Abstract*—Recommender systems are built with the aim to reduce the cognitive load on the user. An efficient recommender system should ensure that a user spends minimal time in the process. Conversational Case-Based Recommender Systems (CCBR-RSs) depend on the feedback provided by the user to learn about the preferences of the user. Our goal is to use the feedback provided by the user effectively by exploiting the interplay among the products to build an efficient CCBR-RS. In this work, we propose two ways towards achieving that goal. In the first method, we utilize the higher order similarity and trade-off relationship among the products to propagate the evidence obtained through user feedback. In our second method, we utilize the diversity among cases/products along with the similarity and trade-off relationship to make the best use of the feedback provided by the user.**

***Keywords*—Preference-based feedback; case-based conversational recommender system; evidence; trade-offs; compromise; diversity**

## I. Introduction

The way product recommendation is handled by a recommender system in most of the on-line websites is very different from how a sales executive would deal with a customer in a store. Though a large amount of data is mined everyday, as rightly mentioned in [5], in some domains where a specific user is expected to buy a product once in a lifetime say for example 'housing domain' or 'a luxury car domain', collaborative systems fail due to lack of data about the user. In contrast to that, knowledge-based recommender systems like the CCBR-RSs are known to handle cold start conditions [6] with ease and work more like their human counterparts. Case-Based Reasoning Recommender systems (CBR-RSs) uses the notion of similarity among cases to approximate the utility of a product to users. Products are represented as feature value pairs. The user is allowed to express their preference by stating one or more feature values. For example, in a camera domain, the user's query may be '10 MP resolution'. A CBR-RS uses the similarity measure to find all products whose resolution is similar to '10 MP'. Sometimes it could be possible that no recommendations are made as the products don't match the query exactly [3]. The advantage with CBR-RS is that even if there are no cameras with exactly '10 MP' resolution, cameras with resolution similar to '10 MP' will be retrieved. A global similarity measure based on feature level similarity is defined using domain knowledge.

A Single-shot CBR-RS stops with recommending products to a user based on their initial query. If the user is not satisfied with the recommendation she has to give a new query to the system. A CCBR-RS helps the user navigate the product space. It engages the user in two phases: a) The feedback phase, where the feedback regarding the products recommended to the user is received, and b) The recommendation phase, where the system recommends appropriate products based on the user query/feedback. We will refer to the set of products recommended in the recommendation phase as the Recommendation set (RS). The system learns the preferences of the user incrementally at every interaction with the user and recommends based on the learned preferences. A CCBR-RS aims to reduce the time a user spends on the system by making the interaction with the user effective and reducing the number of interaction cycles. Traditionally CCBR-RSs reported in literature works on modelling the user in a better way to improve the efficiency of the system. Our work tries to achieve the goal by capturing the rich interplay among the products in the domain.

The authors in Evidence-Based Recommendation (EBR) [2] propose a novel view of the feedback provided by the user in CCBR-RSs. A product in the domain needs evidence that it would be preferred by the user before it could be included in RS. The feedback given by the user plays the role of attributing evidence to each product in the domain. We cannot expect the user to give feedback on each product in the domain, so the evidence from feedback provided by the user on a minimal set of products are propagated to the rest of the products in the domain. We extend the idea of evidence propagation to include higher order propagation by posing the evidence propagation as a random surfer model. The work by authors in [14] focuses on bringing out the advantage in utilizing the trade-off relationship among products. They argue that trade-off is the relationship that is particular of the products in the domain and is not with respect to the user query. McSherry in his work [11] brings out the idea of using compromise (the relationship of a product with respect to the user query) in improving the success rate of a CBR-RS. As the second part of our work, we combine the best of both the static trade-off relationship among products and the dynamic compromise relationship between the user query and the products to bring out the idea of utilizing the collective evidence of the RS and build an efficient CCBR-RS in contrast to EBR which deals with evidence of individual products. We evaluate our methods on three datasets and report the results in comparison to the base works.

In the next Section, we introduce the background for our work: This is followed by the related works in Section III. The details of our approach are presented in Section IV. This is followed by the evaluation and results in Section V and we

conclude in Section VI.

## II. Background

In this section, we go into the details of the background and set the context for our work before we move on to the related works.

### A. Case Representation and Approximation of Utility

We work in the context were the cases/products are from the same domain. For example a camera domain has only cameras in it. Each product, is represented as a fixed set of features/attributes with values corresponding to each of the features. In domains like motorcycle, the features of the domain can be categorized into More is Better (MIB) and Less is Better (LIB) [12]. For example, the feature 'Mileage' is an MIB feature. Given two motorcycles with similar feature values, people often tend to choose the motorcycle with higher 'mileage'. In contrast, 'price' is an LIB feature. Given two motorcycles with similar feature values, people often tend to choose the motorcycle with lower 'price'. If the query is '5000 $' motorcycle, then any motorcycle with the price less than or equal to '5000 $' may be deemed useful to the user. The higher the price above '5000 $' the lower the utility. Had the query been '100 horse power'(MIB feature), then any motorcycle with more or equal horse power would be deemed useful to the user and lesser the value the less useful it is. The feature values are normalised appropriately such that for an MIB feature the highest value is set to 1 and the lowest value is set to 0; for an LIB feature the highest value is set to 0 and the lowest value is set to 1. The score a product would receive is given in equation below where $Q_i$ stands for query value of feature *i* and $P_i$ stands for the value of feature *i* of product *P*.

$$lsim(Q_i, P_i) = \begin{cases} 1, & \text{if } P_i \geq Q_i \\ \frac{P_i}{Q_i}, & \text{otherwise} \end{cases} \quad (1)$$

A global similarity measure based on a Multi-attribute utility theory (MAUT) [7] is used to approximate utility. The similarity of a product to another product is the average of feature level similarities. If Q is the query and P is the product then the similarity between them is given in the equation below, where *lsim* is as given in (1).

$$sim(Q, P) = \frac{\sum_{i \in Attributes} lsim(Q_i, P_i)}{|Attributes|} \quad (2)$$

Among a pair of products A and B, if $A_x$ is greater than $B_x$ for an MIB feature *x*, then product A is said to dominate product B with respect to feature *x*. Similarly, for an LIB feature *y*, product A is said to dominate product B with respect to feature *y* if $A_y$ is less than $B_y$.

### B. Preference based Feedback

The feedback provided by the user in each interaction cycle is used to modify the query for the next interaction cycle. User ratings [15], critiquing [4] and preference based feedback [9] are the common ways in which feedback is obtained from the user. Preference-based feedback (PBF) poses a lesser cognitive load on the user compared to the other feedback mechanisms [16]. In PBF, the user picks one product out of the *k* products recommended to her as her product of preference. The PBF is treated as the query in the next interaction cycle. The various ways in which PBF is used is explained in Section III. We restrict our focus to CCBR-RS with PBF.

### C. Random Surfer Model

In the first part of our work, we use the idea of utilizing higher order relationship among the products to enhance [2]. We solve our problem by posing it as a random surfer model. Assume a hypothetical cyclist Alice who keeps touring with her cycle with no specific destination. In the hypothetical world she lives, all roads are one way, no roads intersect and there is exactly one restaurant at the junction of roads and nowhere else. She keeps travelling on the road and when she encounters a junction of roads, she rests a while in the restaurant and chooses one of the roads randomly and keeps travelling. Occasionally, if she is bored she gets teleported with her bike from a restaurant to some random restaurant and starts touring again. Given a restaurant in her world, what is the probability that she would have visited the restaurant? Let *N* be the number of restaurants, *1-d* be the probability with which she teleports, $inNeighbours_I$ of a restaurant *I* is the set of restaurants such that for each restaurant *X* in the set there is a road from *X* to *I*. $outNeighbours_I$ of a restaurant *I* is the set of restaurants such that for each restaurant *X* in the set there is a road from *I* to *X*. The probability that restaurant *I* would have been visited at time step *t* is given in the equation below. At *t=0*, the probability that a restaurant is visited is *1/N*. The event of the Alice visiting each restaurant can be associated with the flow of evidence to each product in the domain, the details of which we will elaborate in Section IV.

$$p(I; t) = \frac{(1-d)}{N} + d * \sum_{J \in inNeighbours_I} \frac{p(J; t-1)}{|outNeighbours_J|} \quad (3)$$

## III. Related Works

### A. Modelling User Preferences

More Like This (MLT) [9] is an early approach that uses PBF for query expansion. The PBF in the current interaction cycle becomes the query for the next interaction cycle. In MLT all the feature values of the PBF are assumed to be indicative of the preferences of the user. The authors in [9] assert the fact that every attribute value of the PBF may not be equally desirable to the user and they propose weighted MLT (wMLT) that assigns weights to each of the features. The weight is based on the uniqueness of the feature value. For example, if the motorcycle preferred by the user is 'Air Cooled' but the rest of the motorcycles in the RS are 'Liquid Cooled' we can be more sure about the preference of the user on that feature. The feature weights model the preferences of the user and thus helps with the personalisation of the recommendations. The work by authors in [13] points out the drawback with learning feature weights independent of the other features in the domain. For the example considered above, one might have chosen 'Air Cooled' engine because the motorcycle is the cheapest of the lot. Assigning higher weights

TABLE I. PRODUCTS RECOMMENDED TO USER

| | price($) | mileage (kmpl) | top speed (kmph) |
|---|---|---|---|
| A | 5000 | 10 | 120 |
| B | 5000 | 8 | 150 |
| C | 7000 | 6 | 250 |

to 'Air Cooled' engine without considering the cost would not reflect the preferences of the user in an adequate manner. In [13], the authors make an assumption that the utility of PBF is greater than or equal to the utility of each of the rejected products. They use a weighted MAUT based similarity model as given in the equation below to model the utility of a product, where $w_i$ is the weight given to feature *i*.

$$wSim(Q,P) = \frac{\sum_{i \in Attributes} w_i * lsim(Q_i, P_i)}{|Attributes|} \quad (4)$$

Consider the example domain given in Table I. Let us consider an instance where the user is recommended products A, B and C, assume the user preferred A. We make an assumption that the utility of A is greater or equal to the utility of B and the utility of A is greater or equal to the utility of C. The assumptions can be expressed as inequalities. For illustration, the assumption that utility of A is greater than or equal to the utility of B is expressed as an inequality which is given in (5). In practice, the values are normalised and used in the inequalities. The generalised inequality is given in (6). The $\Delta$s represent the difference in the values and $w_i$ represents weight of the feature *i*. The sign corresponding to the weights of LIB attributes is negative as shown in the equations. The inequalities are solved for the feature weights. The number of products recommended to the user is usually lesser than the number of attributes in the domain and hence the solutions to the feature weights form a convex region in the feature weights space. We term the region of solutions as the *preference region* of the user. One solution from the *preference region* is picked as the feature weights.

$$-(5000-5000) \times w_{price} + (10-8) \times w_{mileage} + (120-150) \times w_{topspeed} \geq 0 \quad (5)$$

$$-w_{price} \times \Delta_{price} + w_{mileage} \times \Delta_{mileage} + w_{topspeed} \times \Delta_{topspeed} \geq 0 \quad (6)$$

This model is termed Compromise Driven Preference Model (CDPM). In our work, we do not attempt to model the user by using feature weights.

### B. Profiling Products in the Domain

The products preferred by the users themselves are representative of the preferences of the user. The idea of profiling the products even before the recommendation could start is proposed by authors in [1]. They employ a method similar to the one used by authors in [13] to characterize each product in the domain. Their work argues that each product in the domain has prospective buyers. The prospective buyers of the product are modelled in the form of a region of feature weights in the feature weights space. They assume, for every product, their *k* most similar products as the competitors of the product. A prospective buyer of a product would assume that the utility of the product of her desire is greater than or equal to the utility of each of its competitors.

TABLE II. TRADE-OFF REPRESENTATION AND SIMILARITY BETWEEN TRADE-OFFS

| | price($) | mileage (kmpl) | top speed (kmph) |
|---|---|---|---|
| A | 5000 | 10 | 120 |
| B | 5000 | 8 | 150 |
| C | 7000 | 6 | 250 |
| D | 7500 | 6 | 275 |
| $T_{AB}$ | 0 | 1 | -1 |
| $T_{AC}$ | 1 | 1 | -1 |
| $T_{BD}$ | 1 | 1 | -1 |
| tSim($T_{AB}$, $T_{AC}$) | (0+1+1)/3 = 0.66 | | |

Similar to the work by the authors in [13], inequalities are formed and solved for the feature weights. The solution to the inequalities is a region in the feature weights space which the authors term as *dominance region* of the product. In the course of recommendation, the amount of overlap between the *dominance region* of a product and the *preference region* of the user is used to enrich the measure of utility of a product to the given user. The authors show that using the score from the overlap of the regions improve the efficiency of the CCBR-RS. This work is based on the predicted preference of the user and hence termed Predicted Preference Region-based Method (PPRM). We do not pre-process our data as in this work. No attempt is made to profile the individual products but the theme that the products themselves are representative of the preferences of the user is prevalent in our work too.

### C. Trade-off based Recommendation

Consider the products from Table I. If the user query is 'price: 5000 $' and 'top speed: 200 kmph.' then there is no product in the domain that satisfies all the requirements of the user. If we consider product A, it fails to satisfy the 'top speed' requirement but satisfies the 'price' requirement. Likewise, product C fails to satisfy 'price' but satisfies 'top speed'. Since we are not aware of the compromise a user would be willing to make, say if we are allowed to show only two products at a time then it is desirable to recommend products A and C instead of products A and B. If products A and B are recommended and if the user is not willing to compromise on 'price' then none of the products in the current recommendation cycle satisfies the user. McSherry in [11] proposes to recommend products with varied compromises (this is called Compromise Driven Retrieval (CDR)) to improve the success rate of the recommendation system.

Compromises capture the ability/inability of the products to satisfy the requirements of the user. Even without taking the user query into account, if we compare two products say in our example products A and C, we can notice the distinguishing characteristics of each of the products. The MIB, LIB categorization helps us define the dominance relation between the features of any pair of products. Among the products A and C, A dominates C with respect to the features 'price' and 'mileage' similarly C dominates A with respect to the feature 'top speed'. If one has to choose A over C she has to compromise on 'top speed' for the gain in other features. This relation among the product pairs is termed trade-offs by authors in [14]. They define a representation for the trade-offs and a measure of similarity for a pair of trade-offs.

Table II shows the representation of the trade-offs $T_{AB}$, $T_{AC}$ and $T_{BD}$. A Trade-off represents the set of features one has to compromise for the gain in another set of features in choosing one product over the other. $T_{AB}$ represents that if one has to accept A over B she has to compromise on the feature 'top speed' for the gain in 'mileage'. The dominating feature in the product that one would be willing to accept is represented with a '1' and the dominated feature with a '-1', a feature that neither dominates nor is dominated is represented by a '0'. The similarity between a pair of trade-offs is proportional to the count of the number of matching symbols in the given pair. If T and S are the trade-offs the match/similarity between the trade-offs is given in the equation below. **1**() is an indicator function that gives a value 1 if the symbols match and 0 otherwise.

$$tSim(T,S) = \frac{\sum_{i \in Attributes} \mathbf{1}(T_i = S_i)}{|Attributes|} \tag{7}$$

The authors in [14] propose several methods, one of which uses the trade-off choices made by the user in each interaction cycle to identify the potential products of interest to the user (MLT with Trade-off Matching **MLT TM**). In our work, we combine the benefits of both the *trade-off* based relationship and the diversity based on *compromise* to achieve better efficiency.

### D. Evidence-based Recommendation

In the work by authors in [2], the PBF given by the user in every interaction cycle is used to account for the evidence of the products in the domain. In a given interaction cycle, only *k* products are shown to the user and the user prefers only one product among them, then how is the evidence propagated to other products in the domain? The similarity and trade-off relation among the products is exploited to propagate the evidence of the products. For example, if Jane says she likes 'lemon' better than 'apple' it may be the case that she likes 'orange' better than 'apple' too as both 'lemon' and 'orange' are citrus fruits. Though the feedback given by Jane is only on 'lemon', the similarity relation between 'lemon' and 'orange' helps us to propagate the feedback on 'lemon' to 'orange'. Here in the given example, 'lemon' is the dominant product and 'apple' is the dominated product. The dominance relation of PBF on the rest of the products in the RS is propagated based on the hypothesis that *the dominance relation between two pairs of products are similar if they involve similar dominant and dominated products, provided the trade-off relations between the pairs are also similar*. For example, among the products in Table II the user prefers product A over product C. It is highly likely that the user would prefer product B over product D as products A and B are similar (dominating products) and products C and D are similar (dominated products) and the trade-off similarity between $T_{AC}$ and $T_{BD}$ is high. We base our work on the idea of EBR and provide significant extensions to the same.

## IV. Our Approach

As discussed previously, the motivation of this work is to explore ideas on how EBR can be effectively used to improve the efficiency of CCBR-RSs. We propose two broader themes in this work. The first idea deals with higher order evidence propagation. The second theme deals with collective evidence of the RS as opposed to the evidence of individual products.

### A. Higher order Evidence Propagation

In a CCBR-RS the preferences of the user should ideally be aggregated across interaction cycles. In methods where the preferences are modelled as feature weights, the weights reflect the aggregated preferences. In EBR, the user given feedback in an interaction cycle ascribes preferences to various degrees to every product in the domain. The evidence for a product is aggregated across interaction cycles by taking all the feedback the user has provided into consideration. Imagine a directed completely connected graph with each product as a node. Each edge represents the action of the user choosing one node (destination) over the other (source). The node preferred by the user is called *dominant* node (destination node) and the node that is preferred over is called *dominated* node (source node). The dominant node and dominated node pair is termed as the dominance relation pair. Each edge is associated with a weight. Consider the nodes A and B. If there is an edge from B to A, the weight of the edge from B to A denotes the probability with which product A would be selected by the user over product B. If we can assign weights to the edges of the graph then we can find higher order dominance relationship among products. Let us term the graph preference graph.

At each interaction cycle, we pose the problem as random surfer model and find the probability that the random surfer would be found at every node in the graph (the probability that Alice would have visited a restaurant). In our work since we assume a strongly connected graph, every node is equally likely to be visited from a given node, but the weights that we assign to the edges plays the role of assigning probabilities that an edge will be traversed by the surfer. We can assume that the roads in Alice's world are of different widths and Alice chooses a road with probability proportional to the width of the road. The modified formulation is given in the equation below, where the probability that a road from restaurant *J* to *I* would be taken is represented as $w_{JI}$. $p(I;t)$ is the probability that a node *I* is visited at time step *t*. We keep finding the probabilities of the nodes for successive iterations in an iterative way. If at a time step, the sum of the changes in the probabilities of the nodes of the graph with respect to the previous time step is below a threshold we stop the iterative process and the nodes/products with the top *k* probabilities are recommended to the user. In our set up, the higher the weights of the edges leading to a node from other nodes the higher the probability that the random surfer would visit that node. If a particular product dominates other products to a greater extent then the probability of that product is expected to be high. Based on (3), we have the following formulation:

$$p(I;t) = \frac{(1-d)}{N} + d * \sum_{J \in inNeighbours_I} \frac{w_{JI} * p(J;t-1)}{\sum_{L \in outNeighbours_J} w_{JL}} \tag{8}$$

***Assigning edge weights:*** In our work we made the weight of an edge depend on three quantities a) the similarity of the destination node to the products already preferred by the user

TABLE III. PRODUCT-PRODUCT SIMILARITY

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 1.0 | 0.5 | 0.6 | 0.7 | 0.5 |
| B | 0.5 | 1.0 | 0.5 | 0.5 | 0.4 |
| C | 0.6 | 0.5 | 1.0 | 0.7 | 0.9 |
| D | 0.7 | 0.5 | 0.7 | 1.0 | 0.9 |
| E | 0.5 | 0.4 | 0.9 | 0.9 | 1.0 |

TABLE IV. TSIM OF PRODUCT-PRODUCT TRADE-OFF WITH TRADE-OFF $T_{AB}$

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 0.0 | 0.6 | 0.5 | 0.4 | 0.7 |
| B | 0.4 | 0.0 | 0.7 | 0.6 | 0.5 |
| C | 0.5 | 0.3 | 0.0 | 0.3 | 0.2 |
| D | 0.6 | 0.5 | 0.7 | 0.0 | 0.3 |
| E | 0.3 | 0.5 | 0.8 | 0.7 | 0.0 |

in previous interaction cycles; b) the similarity of the source node to the user rejected products; c) the similarity of the trade-off one has to make in choosing the destination node over the source to the trade-offs the user has made in all the previous interaction cycles. We form $k-1$ dominance relation pairs by pairing PBF with each of the rejected products in every interaction cycle. Let $UDR$ (User Dominance Relations) be the set of all the dominance relations that we have aggregated from the user through PBF, $UPP$ (User Preferred Products) be the set all the PBFs the user has provided and $URP$ (User Rejected Products) be the set of all products that are dominated by any of the products from $UPP$. The edge weights are computed based on (9), where $w_{SD}$ denotes the weight of the edge from the Source node (S) to Destination node (D), $R_d$ and $R_r$ denotes the dominant product and dominated/rejected product respectively in the relation $R$. We term our method **HEBR** (Higher-order EBR).

$$w_{SD} = \alpha * \left(\frac{\sum_{M \in UPP} sim(M, D)}{|UPP|} + \frac{\sum_{N \in URP} sim(N, S)}{|URP|}\right) + (1-\alpha) * \left(\frac{\sum_{R \in UDR} tSim(T_{R_d R_r}, T_{DS})}{|UDR|}\right) \quad (9)$$

***Illustration**:* Consider domain where we have 5 products A, B, C, D, and E. Let us assume products A and B are shown to the user and the user has preferred A over B. Now we need to recommend products based on the feedback from the user. Let us construct the preference graph from the data we have. The product-product similarity matrix is given in Table III. The trade-off similarity of every pair of edge with the user made trade-off (A over B) is computed and let us assume it is as given in Table IV. The weights of the edges are computed based on (9). Let $d$ = 0.85, $\alpha$ = 0.9 and at $t$ = 0 each product is considered to be equally likely. We need to order the product C, D and E in the domain so that they can be recommended to the user. The scores for the products based on EBR are [0.525, 0.590, 0.455] corresponding to C, D and E. The scores based on HEBR for C, D and E are [0.188, 0.180, 0.194]. HEBR places E in the first place even though products C and D are more similar to the PBF (A) than E is to the PBF. E is highly similar to C and D so the evidence from C and D propagate to E making it highly probable. (We cannot talk just in terms of similarity with PBF alone, but we use it for the sake of clarity.)

***Step by step recommendation process**:* The following are the steps in HEBR:

*Step 1*: The preferences of the user are expressed in terms of query Q. Q has one or more values corresponding to the features in the domain.

*Step 2*: Products are ranked based on (2). The top $k$ ranked products are recommended to the user. The user picks a product as PBF and continues her recommendation process or accepts one of the recommended products.

*Step 3*: PBF is added to the set $UPP$. Every product other than PBF in the RS is added to the set $URP$. $k-1$ pairs corresponding to $k-1$ rejected products are matched with the dominant product PBF and are added to the set $UDR$.

*Step 4*: A strongly connected graph 'preference graph' *G* with all the products as nodes is constructed and the weights of the edges are assigned based on (9).

*Step 5*: The probability of nodes in the graph *G* are updated based on (8) iteratively for several time steps till the sum of the difference in the probabilities of the nodes from one iteration to the next iteration is below a fixed threshold.

*Step 6*: The products with top $k$ probability values are recommended to the user.

*Step 7*: The user picks her PBF and continues the recommendation process (back to Step 3) or accepts one of the recommended products

### *B. Evidence for Successful Recommendation*

A successful recommendation depends on the contribution of individual products as well as the collective contribution of the products. In EBR, the RS is filled with products that have higher evidence from the feedback provided by the user. The utility of the RS is approximated by the sum of the evidence scores of all the constituent products. The interaction among the products in the RS is not taken into consideration in this view. The earliest work from the literature that considers the interplay among products in RS is MLT with Adaptive Selection (MLT AS) [10]. The authors categorize the interaction cycles into 'Refine phase' and 'Refocus phase'. In 'Refine phase' the products are recommended based on only similarity, in 'Refocus phase' diverse products are recommended to the user based on bounded greedy approach [17]. They show that the efficiency of the system can be improved by methodically introducing diversity. Though the similarity of products in the 'Refocus phase' is not maximized, the efficiency of the system is expected to improve by taking into account the collective utility of the RS. We base our work on MLT TM.

We explain how MLT TM can be understood in terms of evidence maximization and enrich the method to include the effect of interaction among the products with the help of the compromise criterion.

***MLT TM in the light of EBR:*** Consider the example products given in Table II. Products A and B have good fuel efficiency but lose out on power. Irrespective of the user query, just by analysing the products, one can characterize the preferences of the user who would buy each of the products. Similar to the idea used in PPRM, a prospective buyer of a sports motorcycle would prefer 'top speed' feature more than the fuel efficiency feature. Instead of using the trade-off relations to learn feature weights, MLT TM uses the features themselves to model a given user and recommend products that would better serve her need. EBR records evidence in the level of products whereas in MLT TM the evidence is aggregated in terms of features. In Table II if products C and B are shown to user and the user prefers product C then the fact that feature 'top speed' is preferred over features 'price' and 'mileage' ($T_{CB}$) is used as evidence for promoting product D because the comparison of D with B also yields the same trade-off 'top speed' over 'mileage' and 'price' ($T_{DB}$). In EBR the dominance relations are used to propagate the evidence. In MLT TM, the trade-offs are used to propagate the evidence among products.

***Evidence of the recommendation set:*** Compromises are subjective to the user. McSherry asserts that the compromises a user would be willing to make is not known in advance and is independent of the feature preferences of the user. CDR aims to improve the success of the system by suggesting products that are diverse with respect to the compromises they offer with respect to the user query. We propose a revised utility approximation in MLT TM that accommodates the interaction among the products in terms of compromises they offer to model the effective evidence of the RS in a better manner. We formulate two methods to realize our idea.

***Greedy Evidence Maximization (GEM):*** In GEM, products are added incrementally to the RS. There are three criteria for inclusion of a product in the RS. The first criterion is similarity. If a product is more similar to the query then it is more evident that it would satisfy the requirements of the user. The second criterion is trade-off. The user preferred product in a given interaction cycle is compared against the rejected products and the corresponding trade-offs are recorded. The hypothesis in MLT TM is that a product would be useful to the given user if it makes similar trade-offs as the PBF with the user rejected products. The trade-off evidence score is given in (10), where *C* is the candidate product that we are evaluating to be included in the RS, Rejected List is the set of products in the RS other than PBF, the user preferred product PBF is represented as *P*.

$$tradE(C,P) = \frac{\sum_{X \in RejectedList} tSim(T_{CX}, T_{PX})}{|RejectedList|} \quad (10)$$

The third criterion is different from the first two criteria. It is based on compromises. We formulate the criterion such that it ensures maximization of the utility of the RS as opposed to individual products. The Compromise Set is the set of features with respect to which the candidate product *C* compromises on the PBF/query. It is as given in (11). The compromise distance in (12) measures the dissimilarity in the compromise sets of a pair of products ($C_1$, $C_2$) with respect to the query product *P*. The compromise evidence *compE* measures the utility of *C* if it is added to the RS. Since the RS is filled incrementally when there is no product in RS the value of *compE* is taken as zero. *compE* measures the average dissimilarity in the compromises made by the candidate *C* with respect to the rest of the products included in the RS. We want the dissimilarity score to be high so that the products included in the RS have varied compromises with respect to the query. The varied compromise choices provided by the products in RS improves the utility of RS as a whole.

$$CS(C,P) = \{i | i \in Attributes\ and\ P\ dominates\ C\ with\ respect\ to\ i\} \quad (11)$$

$$compDist(C_1, C_2, P) = 1 - \frac{CS(C_1,P) \cap CS(C_2,P)}{CS(C_1,P) \cup CS(C_2,P)} \quad (12)$$

$$compE(C,P,RS) = \frac{\sum_{R \in RS} compDist(C,R,P)}{|RS|} \quad (13)$$

Each product in the domain is evaluated based on the combined score of the three criteria as given in (14). The product with the highest *cScore* is included in the RS. The products in RS are included in a greedy manner incrementally rather than finding the set of products that maximizes the cScore of the products in RS.

$$cScore(C,P,RS) = \alpha * sim(P,C) + \beta * tradE(C,P) + \gamma * compE(C,P,RS) \quad (14)$$

***Greedy Evidence Maximization Bounded (GEMB):*** In GEM each product in the domain needs to be evaluated $k$ times as the products are added incrementally. Instead of evaluating all the products in the domain every time before including a product in RS, in GEMB a) we set a bound **B** b) we sort the products based on similarity and trade-off evidence and evaluate only the top **B** products for inclusion in RS. This provides a good speed-up of the system with some loss in efficiency.

***Step by step recommendation process:*** The following are the steps for the two methods:

*Step 1 and 2*: Same as in HEBR

*Step 2a (Only for GEM)*: All the products in the domain are selected for further evaluation.

*Step 2b (Only for GEMB)*: The products in the domain are sorted based on the sum of *sim* and *tradE* scores, the top **B** products are selected for further evaluation.

*Step 3*: The products are evaluated based on *cScore* and the product with the highest score is added to the RS.

*Step 4*: While the RS has less than *k* products repeat Step 3.

*Step 5*: The user picks her PBF (PBF becomes the new query) and continues the recommendation process (back to Step 2a or 2b based on the method used) or accepts one of the recommended products

*Handling nominal attributes:* The nominal attributes are treated in the same way in all the methods proposed in this work (HEBR, GEM and GEMB). In the similarity calculation step, we give a value 1 if the attribute value matches, else we give the value 0. In trade-off representation, if both the products have the same attribute values we assign the symbol '0' else we assign '1'. While considering for compromises, if the attribute values are same then we consider it as 'no compromise' else it is assumed to have compromised.

## V. Evaluation and Results

We evaluate our methods with the standard evaluation procedure followed in CCBR-RSs literature [8]. It is widely used in CCBR-RS literature ([8], [9], [13], [1], [14]). We take three real-world datasets namely Camera, PC and Used Cars , each of which has 210, 120 and 956 cases respectively. All the datasets have numeric and nominal attributes. The numeric attributes of our domain are categorized into MIB or LIB features. In Camera domain, among numeric attributes 'price' and 'weight' are categorized as LIB and the rest are categorized as MIB. In Used Cars dataset, the attributes 'price' and 'miles' are categorized as LIB and the others are categorized as MIB. In PC dataset 'price' is the only LIB attribute.

### A. Leave-one-out Evaluation Procedure

We simulate an artificial user. We assume that one of the products in the domain ideally suits all the preferences of the user. The ideal product is removed from the domain (left-out product). We identify a product in the domain that is most similar to the left out product and set it as the 'Target product'. The left-out product is used to generate the initial query for the artificial user. A randomly chosen subset of the attribute values of the left-out product is made as the initial query of the user. We generate three levels of queries namely hard, medium and easy. The level of hardness is based on the number of attribute values included in the initial query. A hard query has only 1 attribute value, a medium query has 3 and an easy query has 5 attribute values of the left-out product.

In feedback phase, if the artificial user encounters the 'Target' product in the RS the recommendation process stops and the number of interaction cycles taken to identify the product of interest is recorded else the product that is most similar to the left-out product is selected from the RS as the PBF. The PBF becomes the query for the next interaction cycle. The product to be left out is selected randomly from the domain and the Target product is fixed, the 3 levels of queries are generated from the left-out product and the recommendation process is simulated using the artificial user. This process is repeated 1000 times in each of the domains used for evaluation. The average number of cycles taken to reach the 'Target' product is calculated separately for each of the levels (hard, medium and easy) of queries. We compare HEBR against EBR to evaluate the effect of higher-order evidence propagation. The GEM and GEMB methods are compared against MLT TM to evaluate the effect of maximizing the evidence of RS considering the interaction among products. We have also compared our methods against MLT and MLT-AS.

[1]http://www.mycbr-project.org/download.html

### B. Results

Tables V, VI and VII display the comparison of HEBR against EBR along with MLT and MLT AS. Tables VIII, IX and X display the comparison of GEM and GEMB against EBR along with MLT and MLT AS. The 1000 queries in each query level (hard, medium and easy) are split into 10 partitions of 100 queries per partition and the difference in averages are tested for statistical significance (paired t-test with $p < 0.05$). The results in bold are significantly better than the rest.

TABLE V. Efficiency in PC dataset (the lesser the average cycle length the better)

| Query Size | MLT | MLT AS | EBR | **HEBR** |
|---|---|---|---|---|
| 1 | 8.29 | 6.09 | 4.08 | **3.76** |
| 3 | 6.14 | 4.22 | **3.20** | **3.28** |
| 5 | 3.67 | 2.19 | **1.97** | 2.31 |

TABLE VI. Efficiency in Camera dataset (the lesser the average cycle length the better)

| Query Size | MLT | MLT AS | EBR | **HEBR** |
|---|---|---|---|---|
| 1 | 11.41 | 6.90 | 5.13 | **4.72** |
| 3 | 9.54 | 5.89 | **4.64** | **4.51** |
| 5 | 6.42 | 4.04 | **3.59** | 3.83 |

TABLE VII. Efficiency in Car dataset (the lesser the average cycle length the better)

| Query Size | MLT | MLT AS | EBR | **HEBR** |
|---|---|---|---|---|
| 1 | 24.42 | 14.32 | 9.28 | **7.68** |
| 3 | 19.18 | 10.91 | **7.55** | **7.45** |
| 5 | 15.12 | 8.08 | **5.85** | 6.58 |

TABLE VIII. Efficiency in PC dataset (the lesser the average cycle length the better)

| Query Size | MLT | MLT AS | MLT TM | **GEM** | **GEMB** |
|---|---|---|---|---|---|
| 1 | 8.29 | 6.09 | 5.50 | **4.72** | **4.20** |
| 3 | 6.14 | 4.22 | 3.96 | **3.52** | 3.31 |
| 5 | 3.67 | 2.19 | 2.19 | **1.99** | 2.01 |

TABLE IX. Efficiency in Camera dataset (the lesser the average cycle length the better)

| Query Size | MLT | MLT AS | MLT TM | **GEM** | **GEMB** |
|---|---|---|---|---|---|
| 1 | 11.41 | 6.90 | 6.28 | **4.99** | **5.00** |
| 3 | 9.54 | 5.89 | 5.45 | **4.66** | **4.64** |
| 5 | 6.42 | 4.04 | 3.94 | **3.41** | 3.64 |

TABLE X. Efficiency in Car dataset (the lesser the average cycle length the better)

| Query Size | MLT | MLT AS | MLT TM | **GEM** | **GEMB** |
|---|---|---|---|---|---|
| 1 | 24.42 | 14.32 | 12.14 | **10.47** | 11.57 |
| 3 | 19.18 | 10.91 | 9.64 | **8.15** | 9.26 |
| 5 | 15.12 | 8.08 | 7.53 | **6.20** | 7.07 |

### C. Discussion

HEBR performs better than EBR for hard queries in all the three datasets but on easier queries, the higher order evidence propagation degrade the efficiency. The performance of HEBR

on medium level queries seems to be on par with EBR. The number of cycles needed for hard queries is high so with more feedback the efficiency improvement is high, in contrast with lesser feedback the noise from the preference graph dominates the feedback leading to the depreciation in efficiency. For hard queries, HEBR reduces the cycle length by 7.8%, 7.9% and 17% in PC, Camera and Used Car datasets, respectively.

GEM performs better than MLT TM in all the datasets for all the query sizes. The performance of GEMB is found to be statistically better than MLT TM in PC and Camera datasets, in Used Car dataset alone GEMB performs only as good as MLT TM. In Used car dataset alone, though the averages of GEMB seems to be better than MLT TM they are not significantly better. The time GEMB takes for execution is in the order of MLT TM but with better performance. If the execution time is critical then GEMB offers marginally good performance in a significantly lesser time when compared against GEM. GEM outperforms GEMB in most of the cases but takes more time to execute when compared against GEMB. Using GEM the reduction in average cycle length for hard, medium and easy queries in PC dataset is is 14%, 11% and 9% respectively, in Camera dataset it is 20%, 14% and 13% and in Used Car dataset the numbers are 13%, 15% and 17%, respectively.

HEBR is expected to perform better than EBR for all query levels but it only performs well in the hard queries. The propagation of noise could be the reason for the same. We cannot expect the users to make informed decisions throughout the recommendation process. HEBR can be extended to account for the noise in the feedback provided by the user. The strength of CBR-RS lies in its ability to explain why a recommendation has been made. Explanations add to the trust of a recommendation system. The work in its current form fails to utilize the explanatory potential of CBR-RS. Our future work would include explanations along with the recommendations made to enhance the value of the recommendation system as a whole.

## VI. Conclusion

We have proposed two themes as an extension to the EBR. The first one is propagation of higher order evidence. The second explains MLT TM in the light of EBR and deals with handling the interaction among products in the RS to enhance evidence maximization. The works are compared against the state of art in the area. The results suggest the effectiveness of our approach in improving the efficiency of state-of-the-art.

## References

[1] Anbarasu Sekar, and Sutanu Chakraborti, "Towards Bridging the Gap between Manufacturer and Users to Facilitate Better Recommendation", The Thirty-First International Flairs Conference. 2018.

[2] Anbarasu Sekar, and Sutanu Chakraborti, "Show me your friends, I'll tell you who you are: Recommending products based on hidden evidence", Accepted in 27th International Conference on Case-Bases Reasoning 2019, to appear in LNAI Springer, 2019.

[3] Bridge, D., "Product Recommendation Systems: A New Direction", In Proceedings of the Workshop Programme at the Fourth International Conference on Case-Based Reasoning, pages 79-86, Vancouver, BC, Canada, 2001.

[4] Burke, R. D., K. J. Hammond, and B. C. Young, "The Findme Approach to Assisted Browsing", IEEE Expert, 12(4), 32-40. ISSN 0885-9000. 1997.

[5] Burke, R., "Knowledge-Based Recommender Systems", In Encyclopedia of Library and Information Systems, 2000. Marcel Dekker, 2000.

[6] Gediminas Adomavicius, Alexander Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions", IEEE Transactions on Knowledge and Data Engineering, v.17 n.6, pp.734-749. 2005.

[7] Keeney, Ralph L., and Howard Raiffa. "Decisions with multiple objectives: preferences and value trade-offs" Cambridge university press, 1993.

[8] McGinty L. and Smyth B., "Evaluating preference-based feedback in recommender systems", In Artificial Intelligence and Cognitive Science, pp 209-214. Springer 2002.

[9] McGinty L. and Smyth B., "Comparison-based recommendation", In S. Craw and A. Preece, editors, Advances in Case-Based Reasoning, Proceedings of the 6th European Conference on Case Based Reasoning, ECCBR 2002, pp 575-589, Springer Verlag, 2002.

[10] Lorraine McGinty and Barry Smyth., "On the role of diversity in conversational recommender systems". In Proceedings of the 5th international conference on Case-based reasoning: Research and Development (ICCBR'03), pp. 276-290, Springer-Verlag, Berlin, Heidelberg, 2003

[11] McSherry D., "Similarity and Compromise". Case-Based Reasoning Research and Development. Lecture Notes in Computer Science, vol 2689. Springer, Berlin, Heidelberg, 2003.

[12] McSherry, David., "Coverage-Optimized Retrieval", Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, pages 1349-1350. Morgan Kaufmann Publishers Inc. 2003.

[13] Mouli, S. Chandra, and Sutanu Chakraborti. "Making the Most of Preference Feedback by Modeling Feature Dependencies" Proceedings of the 9th ACM Conference on Recommender Systems. ACM, 2015.

[14] Sekar A., Ganesan D., Chakraborti S, "Why Did Naethan Pick Android over Apple? Exploiting Trade-offs in Learning User Preferences. In: Cox M., Funk P., Begum S. (eds) Case-Based Reasoning Research and Development. ICCBR 2018. Lecture Notes in Computer Science, v. 11156. Springer 2018.

[15] Smyth, B. and P. Cotter, "Surfing the Digital Wave, Generating Personalised TV Listings using Collaborative, Case-Based Recommendation", In ICCBR '99: Case-Based Reasoning Research and Development, Proceedings of the Third International Conference on Case-Based Reasoning, pages 561-571, Seeon Monastery, Germany, July 27-30, 1999. Springer Verlag, 1999.

[16] Smyth, B., "Case-Based Recommendation", In The Adaptive Web. pp. 342-376, Springer 2007.

[17] Smyth B. and McClave P., "Similarity vs Diversity", In D. Aha and I.Watson, editors, Proceedings of the International Conference on Case-Based Reasoning, pp 347-361. Springer 2001.