Context Similarity Metric for Multi-dimensional Service Recommendation

Liwei Liu, Nikolay Mehandjiev, and Dong-Ling Xu

ABSTRACT: Recommender systems support online customers by suggesting products and services of likely interest to them. Considering the customer context is believed to produce better recommendations, yet it poses unique challenges. If recommendation is generated through previous ratings, narrowing down the set of ratings to those under the target context will limit their number producing poor quality recommendations. A common approach to improve the quality is to aggregate ratings from a number of similar context segments, but establishing which segments to aggregate by unguided enumerations is too computationally intensive.

In this paper we propose a novel context similarity metric to guide the aggregation process, and show how it can be extended across multiple context dimensions. The metric underpins another contribution: a guided aggregation approach to context-based recommendation. This approach can be combined with traditional recommendation algorithms to improve their prediction accuracy through guided selection and inclusion of data segments for training of prediction models. We demonstrate the effect of our approach on the prediction accuracy of a popular memory-based collaborative filtering algorithm. The metric and the approach are validated using a set of data on hotel service ratings under different contexts. Four sets of validating experiments demonstrate the effectiveness of the approach.

KEY WORDS AND PHRASES: Multi-dimensional recommender system, multiple criteria, contextual information, similarity of context

Introduction

The Internet has become the most effective means to gather and use information from e-commerce participants, helping vendors understand what their customers need and expect [20]. A recommender system is defined as "*any system that produces individualized recommendations*

as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options" [11]. Such systems are often used by e-commerce websites to help consumers make purchasing decisions and hence increase e-commerce sales.

Recent research addresses certain limitations of current recommendation technologies, such as improving understanding of users and items in order to take full advantage of the information in the user's transaction histories and other available data, incorporating contextual information, supporting selection based on multiple criteria, and reducing intrusiveness *etc* [4]. In this paper we focus on incorporating contextual information to improve the accuracy with which we can predict customer preferences, and offer two contributions: a multi-dimensional context similarity metric and a guided aggregation approach to incorporating context within traditional recommendation algorithms exemplified by the collaborative filtering algorithm.

Our focus on context is motivated by its importance in shaping user preferences. Indeed, Adomavicius *et al* [3] state that the degree of incorporation of contextual information in a recommendation method can affect the accuracy of predicting consumer preferences. If we take hotel recommendation as our example used throughout the paper, one of the context dimensions affecting user's selection is the time of the trip. A seaside hotel is more popular in summer than in winter. Another context dimension can be the companion with whom the user plans to travel. Travelling with friends will favor central hotels close to entertainment places, whilst travelling with family and especially with children favors quiet hotels.

A conventional way to consider context is to follow the reduction-based approach [3] and only focus on those purchase and ranking histories that are under the same context as the intended one. This however may result in insufficient training data and hence produce poor recommendations. This can be addressed by extending the training set through aggregating data

from wider range of similar context values, presuming that similar context values yield similar user preferences. Seeking effective ways to aggregate data into a training set is the focus of this paper. We aim to address the limitations of the "unguided" approaches (e.g., [3]), which seek better recommendation results by enumerating different combinations of context values.

We propose a novel context similarity metric to guide the process of selecting and aggregating ratings from similar context values, and demonstrate how it can be extended across a number of context dimensions. The metric is the basis on which we develop a guided aggregation approach to enable context-aware recommendation. We demonstrate how our approach enables recommendations with improved accuracy and reduced computational costs. We offer two theoretical contributions and one experimental contribution as follows:

Firstly, the proposed context similarity metric can calculate which context values are similar to the values specifying the target context for recommendation. We extend a single dimensional metric to make it suitable for a multi-dimensional context space.

Secondly, we propose the context similarity metric to guide the aggregation of the data in a training set when there is no sufficient data for quality recommendation within the target context, avoiding "blind" enumeration whilst retaining the quality of prediction of the reduction-based approach and its compatibility with any traditional two-dimensional recommendation algorithms. In this paper, we demonstrate the effect of our approach by applying it to a well-known collaborative filtering algorithm extended with multiple criteria ratings [1].

Our third contribution comprises the results from a comprehensive validation and testing on the metric and the approach through four sets of experiments upon a set of data containing user rankings of hotels under a number of different contexts and criteria.

The remainder of the paper is organized as follows: we begin with a description of several popular recommendation approaches. Then we discuss different context dimensions and present our metric and how it can be used to calculate the similarity of different types of context values. After that, the proposed approach is presented. Four sets of experiments are then implemented to evaluate the approach. Afterwards, we compare our approach with other related work and discuss their differences. We conclude and discuss the future work at the end of the paper.

Preliminaries

Recommender systems can be classified into four types according to the number of decision criteria and number of context dimensions they consider as their input data [27]. *Criteria* here are usually represented as the users' opinions of different item features (or properties). This contrasts with *context dimensions*, which represent different types of parameters characterizing the circumstances in which the item is to be used. Take hotel recommendation as an example, the location and cleanliness of the hotel are the criteria, while a context dimension is the purpose of your visit, such as visit for work or for leisure.

These four types of recommender systems are: (I) "*traditional*" systems or single-criterion 2-dimensional systems (i.e., only use the single criterion of ratings and the dimensions of user and item), (II) *multi-context* systems or (single-criterion) *multi-dimensional systems* (i.e., add context dimensions to the two core dimensions), (III) *multi-criteria* (2-dimensional) systems (i.e., include multiple criteria information), and (IV) *multi-criteria multi-dimensional* recommender systems (i.e., include multiple criteria and context information). Most of the existing recommender systems on the market are "traditional" [3], though within the research community, a significant volume of further work has been done in extending systems to either multi-dimensional ones or multi-criteria area ones, with only a couple of systems integrating both.

Traditional Recommender Systems

Traditionally, recommender systems are based on a single factor/criterion, which is usually a numerical rating that represents user's preference of the whole item [4, 30]. Two types of entities, users and items, are used for the recommendation, which results in the two classical dimensions, $R: Users \times Items \rightarrow Ratings$. A traditional recommender system is initialized by users' ratings that are either explicitly or implicitly collected. Then for each target user it tries to choose the item which maximizes the estimated utility R of the item for this user [4].

The recommender system can predict a rating which a user will give to an item, or order these ratings and return the most highly rated items as a list of top-N recommendations to the target user [38] [44]. In general, recommender systems use the information of user characteristics, item features, and behavior of similar users [31]. The algorithms for making the recommendation are usually classified (e.g. in [4], [19], [23], [18]) into three categories: collaborative filtering, content-based and hybrid algorithms.

Among these algorithms, *collaborative filtering* (CF) algorithm is the most popular one [34] , which recommends to users those items which are liked by similar users. Users are similar if they have similar past rankings, assuming that users who had common interests in the past, tend to have similar tastes in the future [6, 8]. GroupLens, Ringo and Amazon.com *etc* are all successful applications of the CF algorithm [35, 42]. According to Breese, Heckerman and Kadie [10], CF algorithms can be grouped into two classes: memory-based which recommend based on the preference of nearest neighbors, and model-based which recommend by building a model of user ratings. Both practical experience and research have reported that memory-based algorithms such as the nearest-neighbor algorithm have excellent performance in terms of accuracy [44].

The genetic memory-based algorithm, also used in this paper, computes a prediction by aggregating the ratings of other users for the same item, such as:

$$r_{u,i} = \overline{r_u} + k \sum_{u' \in U} sim(u, u') \times (r_{u',i} - \overline{r_{u'}}) \quad (1)$$

Where U denotes the set of the best similar users to our target user u who have rated the target item i; $r_{u,i}$ is the rating given by user u to item i, $\overline{r_u}$ is defined as the average ratings of all the items rated by user u, and k is a normalizing factor in the equation, calculated as $k = 1/\sum_{u' \in U} sim(u, u')$ [4]. Here $sim(u, u')$ represents user similarity.

User similarity is used to locate the "neighbors" of the active user which is also another key issue in CF [4], *i.e.* those users who share similar taste to the active user according to their preference histories. Pearson correlation and cosine-based techniques are the two used most widely [4] [5]. The values returned by both techniques range from -1 to $+1$ and denoted as $sim(u, u')$ in (1). The greater the returned value, the more similar are these two users.

Multi-criteria Recommender Systems

Recommender systems which consider multi-criteria ratings are known as multi-criteria recommender systems, and many of them engage with Multiple Criteria Decision Making (MCDM) methods [28, 39]. As Manouselis and Costopoulou have stated, MCDM methods can facilitate the recommendation process [28]. The utility function of a multi-criteria systems can be denoted as $R: Users \times Items \rightarrow R_0 \times R_1 \times ... \times R_k$, where R_0 represents the total rating if there is any, and R_c represents the rating of each possible criterion c ($c = 1,2, ..., n$). However, MCDM methods (e.g., AHP [36] or multi-attribute utility function approaches [9]) are aligned with situations where the decision is of high value, and users have sufficient time to select criteria weights and rank products according to each individual criterion. Since recommender

systems usually aim to provide "instant" result, incorporating multiple criteria within an effective recommender system would need innovative mechanisms to achieve the speed expected.

One way to leverage multi-criteria ratings into similarity computation is to use multi-dimensional distance metrics [1]. It starts by calculating the distance between two users' ratings for the same items. Euclidean distance is chosen as it is a widely accepted way to compute distance in multi-dimensional space. The inverse of the distance is used to measure the similarity between these two users [1]. The distance between user u and u'on item i is:

$$d(u,u') = \frac{1}{|\ I\ |}\sum_{i\in I}\sqrt{\sum_{j=0}^{n}(r_j - r_j')^2} \tag{2}$$

Where I is the set of items that are rated by both u and u', n is the number of the criteria, while r_j is the rating of user u on criterion j. Since similarity of two users is opposite to the distance of them, we denote the similarity as the inverse of the distance, and it is shown as:

$$sim(u,u') = \frac{1}{1 + d(u,u')} \tag{3}$$

The larger the similarity measure, the more similar the two users. Multiple criteria ratings provide more information about user preferences regarding different aspects of items than single-criterion ratings. Hence multi-criteria recommender systems have the potential to increase the prediction accuracy of user preferences comparing to traditional ones [1] [2]. In fact, there is much research work reporting that multi-criteria recommender systems outperform the traditional ones in most cases, such as in Adomavicius [1] and Zhang [46] *etc*.

However there are also cases reported in literature that multi-criteria recommendations actually underperform traditional single-criterion ones [1]. Even with multi-criteria ratings,

different approaches may provide different levels of accuracy depending on the domain, and one approach may fit one dataset better than the other one [1].

In our earlier work [26], we have demonstrated that the multi-criteria ratings approaches provide a more accurate prediction than a single rating CF in our dataset, especially when the user similarity is calculated through Euclidean Distance. These findings motivate our use of Euclidean Distance for calculating user similarity in this paper.

Another extended area is multi-dimensional recommender systems, where context dimensions are added to the traditional two dimensions to produce better estimates of the utility function R.

Multi-dimensional Recommender Systems

A multi-dimensional, or context-aware recommender systems extend the traditional ones by incorporating contextual information, adding the contextual dimensions to the traditional two dimensions $S = Users \times Items$, resulting in multiple dimensions $S = D_1 \times D_2 \times \cdots \times D_n$. The rating function is then $R: D_1 \times D_2 \times \cdots \times D_n \rightarrow Ratings$ [3].

In the example of recommending movies, contextual information such as time, place, and accompanying persons will be considered, indeed the movie suitable for watching with one's partner in cinema on Saturday is different from the one watched at home with his/her parents on a weekday. Then the recommendation space will become $S = User \times Movie \times Time \times Place \times Accompany$ from the original two dimensions [3].

We will now describe the "context" concept from our perspective and motivate the inclusion of context information in the recommendation.

Context

Context is defined in a variety of ways from literature, such as [40] [24]. Here we use one of the most referenced context definitions, where context is defined as "any information that can be

used to characterize the situation of an entity", and the entity is defined as "a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves" [15].

Context plays an important role in recommendation since it can affect the user's decision. For instance, in the hotel rating example from the introduction of this paper, business travelers would choose an expensive hotel to ensure good rest, whilst leisure travelers may choose a cheaper hotel. Lilien *et al* [25] point out that "consumers vary in their decision-making rules because of the usage situation, the use of the goods or services (for family, for gift, for self) and the purchase situations (catalog sale, in-store shelf selection, salesperson aided purchase)" [25]. Stewart and Malaga [43] argue that the context created by an organization's positioning in a particular set of web links affects the user's trust in the organization after their visit of the web site, and trust is crucial to the success of e-commerce [43]. Adomavicius *et al* state and prove that a more accurate prediction of the user' preferences depends on the degrees of incorporation of contextual information into a recommender system [3], since contextual recommendation can affect customer's purchase behaviors and their trust in provided recommendations [16]. Palmisano *et al* [32] provide a rare example of experimental support by conducting an empirical study that systematically investigates the extent to which contextual information matters in predicting customer behavior. They have built two models, with and without contextual information, demonstrating that the predictive performance with context is higher than that without context for most experimental settings. User preferences may differ depending on the context, but also, more importantly, similar users tend to prefer the same item within the same context or across similar contexts. We follow the same experimental design to validate the value of context within our approach and domain.

The Reduction-based Approach

Context-aware recommendation approaches are classified into three types based on the ways of incorporating contextual information into the recommendation process: *pre-filtering*, *post-filtering* and *contextual modelling* [33]. The difference between the first two approaches is when to include context information, before recommendation or after recommendation. However, there is no clear outperformance between pre-filtering and post-filtering approaches [33], so we use the reduction-based approach [3] as a typical example of the pre-filtering approaches.

The reduction-based approach [3] reduces multidimensional recommendations problem to a traditional two-dimensional problem. For example, in a three dimensional space $S = User \times Item \times Time$, the prediction function can be expressed as a function in a two dimensional space $S' = User \times Item$ where the context dimension $Time = t$ (assuming that we are interested in predicting the ratings at time t)[3]:

$$\forall (u, c, t) \in U \times C \times T,$$
$$R^{D}_{User \times Item \times Time}(u, c, t) = R^{D[Time=t](User,Item,Ratings)}_{User \times Item}(u, c) \quad (4)$$

The dataset D contains records $(user, item, time, rating)$ linking user ratings with a particular moment in time, or a particular value for the target context. According to Adomavicius *et al* [3], a context segment is one such context value or the combination of the contexts values along different contextual dimensions that can impact customer ratings in a significant way. The context segment concept can be also extended to dimensions which use continuous numerical values, *e.g.*, temperature, where the boundaries between segments could be arbitrary and fuzzy.

One of the advantages of the reduction-based approach is that all the traditional recommendation algorithms can be applied without modifications. Indeed, the multidimensional

recommendation space is converted to two-dimensions $S' = User \times Item$, keeping only those ratings which pertain to the target context (e.g., time moment t) for the recommendation [3]. However, by reducing the data available for training to the data from the target context segment only, the quality of recommendations can suffer, and the reduction-based context-aware approach can indeed underperform traditional algorithms such as CF [3].

The solution proposed by Adomavicius *et al* [3] is to combine the reduction-based approach with the traditional algorithm (CF as an example) by using the level of data specificity which produces best results. Data from other context segments is merged into the data from the target one only when this would improve the recommendation. This does indeed produce good quality of prediction by design, however the process of enumerating combinations of context segments is not guided and so is computationally inefficient, especially when there are multiple context dimensions involved. This unguided approach is therefore not suitable when there is only limited computational power. As an alternative, here we propose a context similarity metric to guide the context segments aggregation overcoming the limited data in the target context segment.

Context Similarity Metric

In this work, we propose a guided aggregation approach for recommendations, starting with our work on context similarity along a single dimension [27], and then extending this into a multi-dimensional similarity metric, deriving a single value for similarity between context segments.

Similarity of Context along a Single Dimension

Chen [13] states that different context types require the use of different quantifiable measures of the similarity between two context values. In this paper, we classify context values into three types, *scale, ordinal* and *categorical*.

Scale and ordinal context types allow a simple rule - the closer the context values, the more similar the context segments. For example, under the contextual dimension *temperature* in Celsius, our target user may consider using a service within the context segment where the value is 25°**C**. We call this the "active" context segment. Then the ratings provided under the context segment of 20°**C** are closer to the active segment than the ratings provided under the segment of 35°**C**. To measure the similarity of the two context segments along the temperature dimension, we use the inverse of the numerical distance between the two context values. Thus we use the *context similarity* formula below to identify the "nearby" segments of an active context segment.

$$sim(c, c') = \frac{1/d(c, c')}{\sum_{c' \in C} 1/d(c, c')} \quad (5)$$

where $d(c, c')$ is the distance (say the absolute value of the subtraction of the two values c and c') between the active context segment and a "nearby" context segment. c' presents a "nearby" context segment to the active segment c, and C denotes all the "nearby" context segments which we have selected for our calculation. In our example, the active context segment is 25°**C**, so the similarity between the 20°**C** context segment and the active context segment is 2/3, and the similarity between the 35°**C** context and the active context segment is 1/3. Contrary to the context distance, the context similarity is better with a larger value.

When a context dimension is of the categorical type, we can still calculate context similarity and distance using the following method. We start by organizing the values taken by this context type in a specialization taxonomy. The method for constructing the taxonomy will differ according to the nature of the context dimension, yet it would generally involve integration of existing ontologies and the output of statistical clustering techniques as illustrated below for the *TravelledWith* taxonomy in the hotel recommendation example. The output is a hierarchical tree

which organizes context values according to the manner in which they specialize one another, and an attribute-value mechanism to describe crucial properties of each concept [21]. Once the hierarchical structure is established, the similarity of two concepts can be computed by measuring the distance (say by counting the number of the specialization links, or graph edges) from the active context segment to the other one.

Following our hotel recommendation example above, we can build the following hierarchy tree for the context dimension *TravelledWith*, using the context values given by experts, as shown in Fig. 1. The distance between the segments *WithSpouse* and *ExtendedFamily* is 3 based on this hierarchy tree. Using the distance as a value for $d(c, c')$, we can calculate the similarity of all categorical context types under consideration using Equation (5).

[Insert Figure 1 here]

Figure 1. TravelledWith Context Hierarchy Structure

The effectiveness of this metric for categorical values along a single context dimension has been demonstrated in our earlier work [27]. In the next section we extend the metric so that it can be applied to multiple context dimensions.

Multi-dimensional Context Similarity Measure

Suppose there are two context dimensions in our hotel recommendation example, *VisitWasFor* and *TravelledWith* context dimensions. *VisitWasFor* dimension has three values, *ForLeisure*, *ForBusiness* and *Others*. *TravelledWith* has eight values shown in Fig. 1 in cells with black and italic fonts, such as *WithFriend*, *WithSpouse etc*. Thus the *Leisure-Spouse* segment contains all the hotel ratings for which the visit was for leisure and the traveler was accompanied by his or her spouse. It contains a sub set of data which are tagged with the combination of one context

value *ForLeisure* and another context value *WithSpouse*. These two context values are from two different context dimensions, *VisitWasFor* and *TravelledWith*.

There are multiple ways in which a combined similarity metric can be accomplished, and the corresponding similarity counts will be largely arbitrary, such as counting the graph edges of a context hierarchy tree which combines multiple contexts. In our example above, there are two context dimensions, *VisitWasFor* and *TravelledWith (denoted as V and T)*. Both are categorical, and each context dimension has its own similarity tree (see Fig. 1 and Fig. 2). The direct combination of these two trees is meaningless.

[Insert Figure 2 here]

Figure 2. VisitWasFor context structure

In the solution proposed here, each multi-dimensional context segment is presented in a vector format. In this example, we have two context segments, $C_1 = (V1, T1) =$(*forLeisure, withSpouse*), and $C_2 = (V2, T2) =$(*forBusiness, withOthers*). The distance between these two context segments is calculated in a two-step process:

Firstly, subtract the two vectors C_1 *and* C_2 , where each element in the result vector $D_{1,2} = (d_V , d_T)$ represents the distance between the corresponding context values within each context dimension ($d_V = \Delta V , d_T = \Delta T$). For example, ΔV is calculated by counting the edges from one single segment (i.e., $V1$) to another segment (i.e., $V2$) under its respective context hierarchy tree. ΔT can be calculated in a similar way. Alternatively, it can be calculated by counting the number of nodes between the two segments corresponded nodes (i.e., the nodes number between the node of segment $V1$ and the node of segment $V2$) in the hierarchy tree. Suppose there are m nodes in between, $\Delta V = m + 1$. Bigger value of ΔV indicates less similarity between $V1$ and $V2$.

Following the calculation, the distance between C_1 = (*forLeisure, withSpouse*) and C_2 =(*forBusiness, withOthers*) is $D_{1,2} = (\Delta V, \Delta T) = (2,4)$.

Secondly, normalize the distance as percentages of the maximal distance within each context dimension. To normalize, we calculate the maximum distance in each context tree (i.e., the maximum distance in *TravelledWith* context tree is 6), and then each element in the subtracted vector is divided on the maximum distance within its context tree. The maximum distance vector in our example is denoted as $(max\Delta V, max\Delta T)$. The normalization the distance within d_v dimension will be $\Delta V / max\Delta V$.

Following the two steps above, we are able to calculate the distance of the distance vector:

$$D(C_1, C_2) = \sqrt{(\frac{\Delta V}{max\Delta V})^2 + (\frac{\Delta T}{max\Delta T})^2} \quad (6)$$

The larger the distance between two context cells, the less similar they are to each other. The distance between the two context segments in the example above is $\sqrt{(2/2)^2 + (4/6)^2}$ according to our formula. And the similarity is denoted as the inverse of the distance,

$$Sim(C_1, C_2) = \frac{1}{1 + D(C_1, C_2)} \quad (7)$$

When there are more than two context dimensions involved, an *n*-dimensional vector will be used instead of two-dimensional one.

Having calculated the similarity of context segments and the similarity of users based on their multiple criteria ratings, we can guide the way in which past ratings can be aggregated to produce good recommendations. This metric can help improve prediction accuracy by dividing the original data into segments in terms of their similarity and then guiding the aggregation of only some of the segments depending on their similarity to the target segment.

Recommendation Generation through Guided Aggregation of Ratings

Once we have calculated context similarity, we can address the issue of insufficient rating data within certain context segments by using our guided aggregation approach. Our approach addresses a tradeoff between similarity of context and the number of available ratings determining the quality of predictions. Then we proceed to explain the recommendation process informed by similarity of context which extends the conventional CF algorithm as an example.

Stage 1: Grouping Data

It is presumed that not all contextual dimensions impact customer ratings in a significant way. Statistical methods (*e.g.*, t-test) can be applied on the total ratings of all users to determine whether there is a significant difference among the average ratings in these single dimensional context segments [3]. Only those context dimensions which have significant effect on total ratings will be taken into consideration in the further computation.

In a context-aware recommender system, we are interested in the relationship between ratings provided by users and the combination of the values of the significant context dimensions under which the rating was given. This suggests that we should group the ratings according to context segments. And the prediction of how a user likes an item under a certain context segment can be estimated by the ratings under the same context segment provided by this or other users. However, these could be insufficient for reliable predictions.

Stage 2: Aggregating Segments

Suppose the active context segment c has insufficient data for reliable predictions, we can then aggregate the data from other similar context segments c'. A core assumption of our approach is that the larger the similarity $sim(c, c')$ between the active and the other context segment, the

closer the similarity in user ratings, and so the predictions based on the data from a similar context segment c' tend to be better than predictions from a more distant one. We thus use the similarity metric between context segments to guide the aggregation process and avoid the computational complexity associated with "blind" enumeration of different context segments. In addition, the number of extra data "borrowed" should be more than the number of the original data within the active context segment to ensure adding up data for a more accurate prediction. Since if the "borrowing" data are less than the data in the active segment, there is a high probability that many more users are added with limited number of ratings, which will make the data sparser and cause less accurate predictions.

Stage 3: Terminating Aggregation

The question is when to stop aggregating, if the data from the closest context segment is still not sufficient for quality prediction. As we decrease the similarity threshold and use ratings from context segments further away, we increase available data, yet the quality of predictions may decrease because those context segments would inform different kind of decisions compared to the active context segment. To establish the optimal similarity threshold, we have devised a simple algorithm with linear complexity suitable for situations of limited computational power:

Predict ratings from active context segment c;
Calculate prediction error from c
If predictor error is too large then
Do
  Find c′ for which $\mathrm{sim}(c, c')$ is maximum
  Aggregate data from c' into c if the number of data in c' is no less than in c
  Calculate prediction error from c
Until prediction error starts to increase or c includes all data

Note that it is theoretically possible that the lowest prediction error is achieved when all context segments are included in the prediction, which is also another terminating condition. We estimate the time complexity of our approach as linear $O(n \cdot m)$, where n and m are the number of context values from two context dimensions. Indeed, individual segments will be included in turn according to context similarity from high to low till the whole dataset when necessary.

Stage 4: Generating Recommendations

Recommendations are generated on the principle that the higher the overall predicted rating for an item, the higher the chance for this item to be liked. All users' ratings are stored with their context values. The prediction of what rating a user would give to an item within a certain context segment is estimated by the data associated with the same segment. Any of the traditional rating estimation algorithms can be used upon the segments for the prediction generation. When there are too few data for reliable prediction, the data stored under close segments are "borrowed". Once the overall ratings of items unknown to the target user are estimated, the recommendation can be made by choosing the highest rated item or by the first N items in a descending order of all the estimated ratings, thus bridging the boundary between ratings estimation and top-N list generation [44].

Here, we focus on the conventional CF algorithm since it is one of the commonly used and successful algorithms in recommender systems. Especially the nearest neighbor/memory-based CF (used in this paper) has been reported as an excellent performer by both practical experience and related research [44]. We extend it with multiple criteria rankings to reflect the higher degree of user involvement with services compared to products [37], and hence the more specific demands users place on services.

To validate the effectiveness of our metric whilst controlling for variations introduced by the recommendation approach, we use the memory-based extended CF approach with and without using our metric as guidance, and compare the results from the two configurations.

Approach Summary

Generally, the rating a user will likely give to an item under a context is predicted by using the ratings provided by *similar users* under *the same or similar context segments.* User similarity is calculated through Euclidean Distance over multiple criteria ratings on users' historical ratings. In detail, the prediction of the rating that user u would give to the item i is based on aggregating the similar users' ratings on item i by using Equation (1). Similar users are calculated by considering multiple criteria using Equations (2) and (3).

In a nutshell, the proposed approach extends the conventional recommendation algorithms and produces context-aware recommendations by considering context information and multiple criteria rankings. This approach applies a novel context similarity metric as a guide to optimize the number of data points considered in the prediction. The approach can help improve the prediction accuracy of any extended algorithms without changing their algorithms. It also can help to provide more variety in generating the top-N recommendation list, since different target contexts will result in different ratings' predictions. How to generate accurate recommendation list considering context constitutes an interesting topic for future research.

Validating Experiments

A set of feedback and user ratings data for an example service domain (hotel stays) is collected from a well-known rankings site. Our data set includes the records from several major English-speaking countries between 1999 and 2010. This dataset has information about the context of use

and ratings on five criteria in addition to the total rating. The five criteria are Value, Rooms, Location, Cleanliness and Service. All the individual criteria ratings and the total ratings range from 1 (poor) to 5 (excellent). Apart from these five criteria, there are also three context dimensions: *DateofStay*, *VisitWasFor*, and *TravelledWith*.

This particular data set includes 400,106 records. Since the raw data is too sparse to compute, we aggregate ratings of all the hotels from the same brand with the same number of stars, using the assumed equality between the levels of service quality and the similarity of furnishing standards of such hotels. Even after aggregating, the majority of the users only rate one hotel, and more than half of the hotels are only being rated once as well. We remove the dirty records, and the hotels and users that have too few records due to the requirements in calculating the user similarity in the memory-based CF algorithm. The final dataset has 8,681 records, with 788 users and 509 hotels. The sparsity level is 0.9784 (i.e., $1 - total\ records/(number\ of\ users * number\ of\ hotels)$).

We group all the ratings based on the context values for each dimension, and calculate the average total ratings by each user. Paired comparison t-tests are applied to identify the context dimensions which affect the total ratings significantly [3]. The probability of wrongly rejecting a true null hypothesis *p-value* is used as the measure to support our null hypothesis that there is no difference between the two compared context segments within one dimension. After applying t-test to our dataset, two of the three context dimensions are left for our further experimentation: *VisitWasFor*, and *TravelledWith*.

Experimental Setup

To study the performance of the proposed approach, we investigate if it improves the performance of a well-known CF algorithm. Indeed, our approach is designed so that it can be

combined with any traditional recommendation algorithm to provide more accurate predictions tuned to the context of intended use without changing the algorithm itself. Additionally, to control for the quality of the recommendation algorithm, we do not compare our approach where we extend CF with other generic algorithms. Instead we compare it with the non-extended original CF to test if our extension can improve the prediction accuracy. This follows the experimental design of Palmisano *et al* [32].

The proposed approach is implemented and tested using the dataset (D) mentioned above. The overview of the major steps is as follows. Firstly, the dataset is split into training set (D_t) and test set (D_e) randomly, $D_t \cap D_e = \emptyset, D_t \cup D_e = D$. Multiple pairs of disjoint training and test sets are obtained from the same initial data. Secondly, the proposed approach is conducted on the training set D_t. Different sets of experiments are designed for validating different assumptions on which the approach is based and the effectiveness and efficiency of the approach as outlined next. The detailed design is shown before each set of the experiments as discussed below. Thirdly, the test set D_e is used to evaluate the approach by comparing the prediction generated from D_t with the original records within D_e. This whole process is repeated over each pair of training set and test set. T-tests are applied to the final results for a more general conclusion.

To evaluate key ideas behind different stages of the approach, four sets of experiments are reported. The first set is designed to test our core assumption that considering context similarity leads to better predictions. To validate our approach, the second set tests if prediction improves when "borrowing" ratings data from a similar context segment rather than from a dissimilar one. The third set of experiments tests if prediction accuracy improves by increasing the number of the context segments providing training data. In the last set of experiments we test whether using

the similarity of context segments as a weight in the prediction formula produces more accurate results than when we do not consider this weight.

Our experiments were implemented using MatLab R2009a. All the experiments were run on a PC with Windows XP Prof., Intel Pentium(R) Core(TM) 2 CPU, 2.13GHz and 2GB RAM.

Contextual Structure

As discussed earlier, *VisitWasFor* and *TravelledWith* are considered as influential context dimensions. Next we use specialization taxonomy to build their structure. *TravelledWith* context has eight values (or categories), their definitions are vague and there is overlapping in some of them which makes its structure difficult to define. To build its structure into a hierarchy tree, we firstly gathered subjective information about how context values of *TravelledWith* affect decisions and decision criteria. Eight persons who have knowledge in the MCDM area were asked to fill in a questionnaire while keeping in mind choosing the hotels based on the criteria they are concerned with (e.g., cleanliness, service *etc*) or travelling situation (i.e., the context information). A hierarchy clustering method [14] such as dendrogram, was then used to analyze the questionnaire results about similarity of context circumstances by using SPSS 16.0. The shorter the distance between any two concepts within the dendrogram, the more similar are the corresponding two concepts. Concurrently we used Formal Concept Analysis [45] to create a lattice based on the criteria highlighted as important under different context values. We then constructed the context hierarchy tree through combining the dendogram and the lattice with a third structure - the ontology of Human relation [29]. The resultant *TravelledWith* context hierarchy tree shown in Fig. 1 comprises the clusters from the formal concept analysis, yet it also considers the length difference shown on the graph between context values in the dendrogram.

In Fig. 1, the eight circles with Bold and Italic characters are *TravelledWith* context values: *Solo* (1), *withTour* (7), *withOthers* (8), *withFriend* (3), *withSpouse* (2), *ExtendedFamily* (6), *FamilyWithChild* (4) and *FamilyWithTeenager* (5). The numbers within brackets here are used as a shorthand reference to each value. Fig. 2 is the hierarchical taxonomy structure of *VisitWasFor* context dimension, and *forBusiness* (1), *forLeisure* (2) and *Others* (3) are its three values. For example, $V1T8$ represents *forBusiness-withOthers* context segment.

Evaluation Metric

Mean Absolute Error (MAE) is used as the evaluation metric in this paper. MAE is a measure of the average absolute deviation between a predicted rating and the user's true rating.

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N} \quad (8)$$

N is the number of pairs of real ratings and predictions $< p_i, q_i >$. The lower the MAE, the more accurate are the predictions [38].

Coverage is used as another evaluation metric to measure the proportion of items that can be recommended [41]. We predict for each user/item pair in our test sets, and measure the percentage for which a prediction was provided [17].

Effect of Context Consideration

This first set of experiments tests the assumption of this work, that the inclusion of contextual information can help improve prediction accuracy.

Experiment design

Two training sets are used. The first training set is from $V2T2$, indicating that we predict by using the data from the same context presenting the consideration of context situation. The second one presenting the situation without considering context, is from *MostData*, which has

the records from the combination of several context segments. Those context segments involved are $V1T8$, $V2T2$, $V2T3$ and $V3T2$, chosen since they have larger sizes than the others. At the same time we limit the number of $MostData$ training set records to 3,200. This is larger than the number of records in the $V2T2$ training set, which is 2983. If the prediction result of $V2T2$ is better than the prediction based on the $MostData$ dataset, this indicates that taking into account the context of use would improve the accuracy of the prediction. To test the approach, 10% of the records from $V2T2$ are selected randomly multiple times as different test sets.

Impact of context dimensions

Following our experiment design, 10 test sets are selected from data which are not used in their training sets. We compare the result from each test set with the same training sets each time. The results are showing in Fig. 3 and Table 1.

Table 1. T-Test from Training $V2T2$ and Training set *MostData*

[Insert Table 1]

[Insert Figure 3]

Figure 3. MAEs for testing the impact of context

For all ten test sets, MAEs from the training set of $V2T2$ are better than those from training sets of *MostData*, even though *MostData* training set size is larger than the size of the $V2T2$ training set (generally, the bigger the size of the training set, the better the prediction is). This indicates that the consideration of context does improve prediction compared to the prediction without consideration of context.

The results above are used as samples to estimate the relationship between two populations, which are the MAEs computed using the training set $V2T2$, and those computed using the training set *MostData*. The null hypothesis will be the mean of MAEs from $V2T2$ is greater than

or equal to the mean of MAEs from *MostData* ($\mu_1 - \mu_2 \geq 0$). The alternative hypothesis states that the mean of MAEs from $V2T2$ is smaller than that from *MostData*. T-test is applied to test our statement.

The results in Table 1 suggest we should accept the alternative hypothesis since this is a one-tailed test and the *p-value* is much smaller than 0.05 (the significance level we assumed). In other words, the mean of MAEs from training set $V2T2$ is significantly smaller than the mean of MAEs from training set *MostData*. The results indicate our assumption that context affects users' ratings positively, and that the prediction within only one context dimension is more accurate than the prediction across several context dimensions. This is also supported by the coverage measure where the average coverage of predictions from $V2T2$ is 0.7695, which is much improved compared to the coverage from the *MostData* training set (0.4936).

Effect of Similar Context Prediction

In the second set of experiments, we test whether data from similar context segments can be "borrowed" to predict ratings from an "active" context segment with insufficient number of ratings, and more importantly, if these predictions are more accurate when compared to "borrowing" data from dissimilar context segments.

Experiment design

There are only 74 records in $V2T1$, and 20% of them are selected as a test set. The remaining 60 records of $V2T1$ are not enough as training data to produce a reasonable prediction. In this set of experiments, we compare the prediction accuracy produced by "borrowing" data from similar segments and dissimilar segments. There are two training sets, the combination of $V2T8$ and $V2T3$ (denoted as $V22T83$), and the combination of $V1T2$ and $V3T2$ (denoted as $V13T22$). According to our context similarity metric, $V13T22$ is further away to $V2T1$, than $V22T83$

context segment. In the following, $V13T22$ is named as *FarSegments*, while $V22T83$ is named as *CloseSegments* for the ease of reading. The sizes of both training sets are the same.

Impact of similar context

10 test sets are selected, and the MAEs were shown in Fig. 4. We can see that MAEs from training set *CloseSegments* generally are smaller than those based on training set *FarSegments,* which means that the training set *CloseSegments* provides a more accurate prediction. What is more, in *FarSegments*, there are 2 out of 10 records which cannot be predicted, which is a significant proportion. In comparison, *CloseSegments* provides constant smaller prediction errors. From our context similarity metric, *CloseSegments* are closer to $V2T1$ than *FarSegments*. Thus the experiments support our assumption that the closer the two context segments are, the better the prediction is.

[Insert Figure 4]

Figure 4. MAEs for testing whether the inclusion of similar context segments produce more accurate prediction

To test the difference of two populations generally, as what we have done in the previous section, our alternative hypothesis states that the mean of MAEs from *CloseSegments* is smaller than that from *FarSegments*. T-test is applied and the results are shown in Table 3.

[Insert Table 2 here]

Table 2. T-Test from Training set CloseSegments and Training set FarSegments

The resultant *p-value* is small enough to accept the alternative hypothesis that the mean of MAEs from training set *CloseSegments* is significantly smaller than the mean of MAEs from training set *FarSegments.* All of the results indicate that the predictions based on the data from similar context segments are better than those based on the data from further context segments.

Effect of Prediction with Different Number of Similar Context Segments

Our guided aggregation approach "borrows" data from similar context segments to address situations when there is not enough data within the target context segment. The approach recognizes the tradeoff between the degree of aggregation (the number of context segments that is involved in the aggregation) and the prediction accuracy, and attempts to calculate the optimal number of context segments to use. This set of experiments aims at validating this approach by testing the relationship between the degree of aggregation and the prediction accuracy.

Experiment design

20% of $V2T1$ records are selected as the test set, as $V2T1$ is a small segment where lack of training data is an issue. We are "borrowing" data from other similar context segments one by one, from the closest segment to the whole dataset at the end. As a result, the similarity of the context segments is gradually reduced following the increase of the segments borrowed. However, we limit the size of the training set to reduce the effect on accuracy caused by oversized training set.

Impact of involved context segments

Following the experiment design, 10 test sets are selected randomly from $V2T1$. The training set initially has only 60 records. We "borrow" data from other context segments from the closest to the furthest, finally including the whole set. The final number of records is limited to 500 to demonstrate the viability of our approach in situations of limited computation power. The average of MAEs under each segments combination is plotted. The similarity of a combination of segments is calculated by averaging the similarities of all the involved segments, each calculated through our context similarity metric.

As seen in Fig. 5, using the remaining records from $V2T1$ for prediction results in a large MAE due to insufficient data in the training set. When we start to "borrow" data from other similar context segments, MAE decreases, indicating an increase in the prediction accuracy. It reaches a minimum value and then it increases again as data from other segments is added, reducing the average similarity of the combination of context segments. This result indicates that extra data from most similar context segments can improve the prediction accuracy. However, if the similarity becomes too low, then the prediction accuracy is not very good either. Fig. 5 shows a slow downturn after the minimum point, which is somehow unexpected and warrants future study. We would suggest considering the balance of the quantity of training data and the prediction accuracy in practice.

[Insert Figure 5 here]

Figure 5. Relationship between quantity of segments and accuracy

Fig. 6 and Fig. 7 show the changes of prediction accuracy and coverage following the increase of the quantity of segments with a controlled training size. Following the increase of similar segments, the training size increases, and the accuracy and the coverage are becoming better as well. At the point of "3segments" (meaning that the 3 similar segments are added), it reaches the controlled size, and the accuracy becomes the best among all the candidates, while the coverage reaches the peak as well. From this point onwards, the relatively small coverage is mainly due to the control of the training size to 500 records.

[Insert Figure 6 here]

Figure 6. Average MAE with increasing segments

[Insert Figure 7 here]

Figure 7. Average coverage with increasing segments

Effect of Prediction considering Context Similarity as Weight

A further use of our context similarity metric would be to value the distance between the active context segment and the other context segments which participate in the recommendation. Thus, this set of experiments will focus on testing whether using context similarity as weight in the prediction formula can improve the accuracy of prediction.

Experiment design

We follow the design of the second set of experiments, and select 20% records from $V2T1$ as test set, and all data from *CloseSegments* and *FarSegments* as two training sets. However, in this set of experiments we use context similarity as weight for prediction. The prediction formula is based on extending the CF prediction function as follows:

Suppose we are predicting the rating r that user u will provide for item i under the context segment c. The rating $r_{u,i}$ is calculated by:

$$r_{u,i} = \bar{r}_u + k \cdot w \cdot \sum_{c' \in C} \sum_{u' \in U} sim(u, u') \times (r_{u',i} - \overline{r_{u'}}) \times sim(c, c') \qquad (9)$$

There are two normalizing factors in the formula, k and w, $k = 1/\sum_{u' \in U} sim(u, u')$. $sim(u, u')$ is the similarity between user u and the other user u' who has provided ratings in the same context segment, in our case, in the context segment of c. The second normalizing factor (or the weight calculated from similarities of the segments) is $w = 1/\sum_{c' \in C} sim(c, c')$ where $sim(c, c')$ is the similarity between context segment c and another segment c'. Here C includes the similar context segments that are involved in the prediction. The more similar the segments, larger the weight and the larger the role that $(r_{u',i} - \overline{r_{u'}})$ play in prediction.

Impact of context weight

We predict the test sets using the *CloseSegments* and *FarSegments* training sets following Equation (9). The results are shown in Fig. 8.

[Insert Figure 8 here]

Figure 8. MAEs for testing whether context weight matters

Generally speaking, the MAEs are smaller when considering context similarity as weight than without considering it in final prediction. When we apply T-tests to test whether the MAEs with context similarity weight is smaller than those without it (alternative hypothesis), the *p-values* of both tests are greater than 0.05. Thus, T-tests do not support the conclusion that using context distance as a weight can help improve the accuracy of the prediction, although the average of MAEs with context weight is smaller than those without using context weight in this set of experiments. At the same time, the average of the coverage from the predictions considering context weight is smaller than that from predictions without considering context weight due to the further control of the selection of user neighbors in the former. This is also the area we would like to explore further in order to maximize the use of our context similarity metric and to tune our approach in the future.

Related Work

The majority of research in extended RS discussed earlier addresses multi-criteria RS [1, 2, 22, 26, 28]. To the best of our knowledge, there are three pieces of work which we are aware of reporting work similar to ours.

Chapphannarungsri and Maneeroj [12] have developed a multi-criteria and multi-dimensional recommender system for movie selection. They distribute the overall rating to each criterion as the criteria ratings for the multiple criteria aspect. In the multiple dimensional space, multiple

linear regression is applied for the multiple context modeling to avoid the data loss problem caused by the reduction-based approach [12]. However, the multiple linear regression is carried out for each user in this paper. When a user has rated a very limited number of movies, the coefficients of each context dimension can be uncertain.

Our context similarity metric can help solve the data scarcity problem caused by reduction-based approach proposed by Adomavicius *et al* [3]. In their work, they point out that once reduction-based approach is applied to preparing the data, this results in a fewer data points in the training set and so the results can be worse compared to predictions based on the whole data set without taking context into consideration. To overcome this problem, they compare the prediction under a certain segment with the one under the whole data set. They keep only the segments that produce better recommendation results (outperforming segments), and the predictions based on those segments are used for the final recommendation. Thus, all segments except the very small ones are enumerated and recommendations based on each are generated and compared to establish which the outperforming ones are.

The process of this enumeration is not satisfactory under conditions of limited computational power. The computation complexity of the calculations for each segment is an issue, since recommendation calculations are quite time-consuming. In our context example with test set from a segment $V2T1$, its prediction accuracy is worse than CF prediction as shown in Fig. 6. Thus we have to try other segments. There are different ways to combine segments, such as combining according to *TravelledWith* context dimensions, or according to *VisitWasFor*, or both of them. When all the segments are quite big, the calculation needs to search over the combination of all the possible segments i.e., the target segment and all the segments within its supersets. The time complexity of including segments following this approach is $O((1+$

$\sum_{i=1}^{n-1} C_{n-1}^{i})(1 + \sum_{k=1}^{m-1} C_{m-1}^{k}))$ when there are two context dimensions (one with n context values, the other one with m context values). This compares unfavorably with our approach, where, as stated earlier, the time complexity is linear $O(n \cdot m)$. The more context dimensions are involved, the bigger will be the advantage of our approach compared to the unguided one. Empirically, the time that the aggregation process takes in our approach is so short that it can be ignored in comparison with the time of the other processes such as the calculation of user similarity, *etc*. In our particular experiment, the aggregation process takes less than 0.0001s. This is not true for the unguided aggregation approach, where at the initial pass recommendation rankings have to be generated on the basis of each context segment to establish the outperforming ones.

To the best of our knowledge, so far there is only one paper proposing a context similarity metric. Chen [13] presents a context-aware CF recommender system to predict a user's preferences in different context situations. He defines a method to calculate the similarity of context types. The method uses Pearson Correlation between two different context variables with respect to their ratings to compute the relevance of two contexts, and then the similarity returned will be used as weight for recommendation prediction [13]. This method requires a significant number of ratings by users who have rated the same items under different context segments. Thus it does not suit some datasets, such as our rating dataset. The number of the people who rate the same hotel at least twice under different contextual dimensions is too limited to calculate the context similarity. This is also the reason why we do not compare their metric with ours.

Discussion and Conclusion

This paper presents our novel approach to incorporating context in multi-criteria RS, which has resulted in a recommendation method covering both criteria and context dimensions. To address

the potential problem of scarcity of data encountered by existing reduction-based approaches, we introduce a metric for calculating context similarity within multi-dimensional context spaces. To measure similarity between ordinal and scale values we use the inverse of normalized distance, for categorical context values we first have to structure the context dimension by combining two clustering techniques and concept specialization taxonomy, producing a context specialization hierarchy. This allows us to use simple metrics such as the number of specialization links between the concepts as reported here, or the size of the concept difference calculated using description logic as detailed elsewhere [7]. Extending context similarity to multiple context dimensions allows its use for guiding the process of aggregating ratings from similar context segments when there is insufficient data under the target segment. Our aggregation approach can be used to extend any traditional recommendation algorithm. In this paper we have illustrated this by extending memory-based CF, demonstrating the improvement in accuracy brought about by our extension through a number of experiments using hotel ratings data. The experiments show that (a) using context information produces better recommendations; (b) we can effectively guide aggregation of training data by choosing the most similar context segment; (c) the benefits of aggregating data beyond an "optimal" point decrease since data becomes less relevant; and (d) using context similarity as a weight within the prediction formula does not bring about statistically significant improvement in prediction accuracy.

One of the weaknesses of the approach presented here is that whilst our approach deals with all three types of context dimensions (scale, ordinal and categorical), we experiment only with the most challenging one, the categorical context types. This is due to the lack of the other two types of contextual data in our test set. The other limitation of our approach as described here is estimating the similarity of categorical context values through the simple measure of counting

specialization edges in a context taxonomy. This makes our similarity estimate dependent on the degree of granularity of the concept taxonomy. In our future work we are planning to use semantic reasoning instead for measuring the similarity of categorical context values.

The area of research in multi-criteria and multidimensional recommender systems is to be developed further, and the way in which models and processes are integrated within the recommender systems is to be carefully considered within our further research plans.

REFERENCES

1. Adomavicius, G. and Kwon, Y. New recommendation techniques for multicriteria rating systems. *IEEE Intelligent Systems*, 2007,48-55.
2. Adomavicius, G.; Manouselis, N.; and Kwon, Y. Multi-criteria recommender systems. In R. Francesco, R. Lior, S. Bracha, and B.K. Paul (eds.), *Recommender Systems Handbook*. New York: Springer, 2011, pp. 769-803.
3. Adomavicius, G.; Sankaranarayanan, R.; Sen, S.; and Tuzhilin, A. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems* (*TOIS*), 23, 1 (2005), 103-145.
4. Adomavicius, G. and Tuzhilin, A. 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*, 17, 6 (2005), 734-749.
5. Ahn, H.J. Utilizing popularity characteristics for product recommendation. *International Journal of Electronic Commerce*, 11, 2 (2007), 59-80.
6. Anand, S.S. and Mobasher, B. Intelligent techniques for web personalization. In B. Mobasher and S.S. Anand (eds.), *Intelligent Techniques for Web Personalization*. Berlin: Springer, 2005, pp.1-36.
7. Baader, F. and Nutt, W. Basic description logics. In F. Baader, D. L. McGuinness, D. Nardi, and P.F. Patel-Schneider (eds.), *The Description Logic Handbook: Theory, Implementation, and Applications*. London: Cambridge University Press, 2003, pp. 43-90.
8. Balabanović, M. and Shoham, Y. Fab: content-based, collaborative recommendation. *Communications of the ACM*, 40, 3 (1997), 66-72.
9. Belton, V. and Stewart, T.J. Multiple criteria decision analysis - an integrated approach. Massachusetts: Kluwer Academic Publisher, 2002.
10. Breese, J.S.; Heckerman, D.; and Kadie, C. Empirical analysis of predictive algorithms for collaborative filtering. *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*. San Francisco: Morgan Kaufmann, 1998, pp.43-52.
11. Burke, R. Hybrid recommender systems: survey and experiments. *User Modeling and User-Adapted Interaction*, 12, 4 (2002), 331-370.
12. Chapphannarungsri, K. and Maneeroj, S. Combining multiple criteria and multidimension for movie recommender system. *Proceedings of the International Multiconference of Engineers and Computer Scientists*. Hong Kong: Newswood Limited, 2009.

13. Chen, A. Context-aware collaborative filtering system: predicting the user's preference in the ubiquitous computing environment. *Location- and Context-Awareness.* Berlin: Springer , 2005, pp.244-253.
14. D.Manning, C.; Raghavan, P.; and Schutze, H. *An Introduction to Information Retrieval.* London: Cambridge University Press, 2009.
15. Dey, A.K. Understanding and using context. *Personal and Ubiquitous Computing*, 5, 1 (2001), 4-7.
16. Gorgoglione, M.; Panniello, U.; and Tuzhilin, A. The effect of context-aware recommendations on customer purchasing behavior and trust. *The 5th ACM Conference on Recommender Systems*. New York: ACM Press, 2011. pp.85-92.
17. Herlocker, J.L.; Konstan, J.A.; Terveen, L.G.; and Riedl, J.T. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22, 1 (2004). 5-53.
18. Jannach, D.; Zanker, M.; Felfernig, A.; and Friedrich, G. *An Introduction to Recommender Systems*. London: Cambridge University Press, 2011.
19. Karta, K. *An investigation on personalized collaborative filtering for web service selection*. 2005.
20. Kim, K. and Ahn, H. Collaborative filtering with a user-item matrix reduction technique. *International Journal of Electronic Commerce*, 16, 1 (2011), 107-128.
21. Kwon, O. and Kim, M. MyMessage: case-based reasoning and multicriteria decision making techniques for intelligent context-aware message filtering. *Expert Systems with Applications*, 27, 3 (2004), 467-480.
22. Lakiotaki, K.; Matsatsinis, N.F.; and Tsoukiàs, A. Multi-criteria user modeling in recommender systems. *IEEE Intelligent Systems*, 26, 2 (2011), 64-76.
23. Leimstoll, U. and Stormer, H. Collaborative recommender systems for online shops. *Proceedings or the 13th Americas Conference on Information Systems*. New York: Curran Associates, Inc, 2007.
24. Lieberman, H. and Selker, T. Out of context: computer systems that adapt to, and learn from, context. *IBM Systems Journal*, 39, 3&4 (2000), 617-632.
25. Lilien, G.L.; Kotler, P.; and Moonrthy, S.K. *Marketing Models*. New Jersey: Prentice Hall, 1992.
26. Liu, L.; Mehandjiev, N.; and Xu, D-L. Multi-criteria service recommendation based on user criteria preferences. *The 5th ACM Conference on Recommender Systems.* New York: ACM Press, 2011, pp. 77-84.
27. Liu, L.; Mehandjiev, N.; and Xu, D-L. Using contextual information for service recommendation. *The 44th Hawaii International Conference on System Sciences*. Washington, DC: IEEE CS Press, 2010.
28. Manouselis, N. and Costopoulou, C. Analysis and classification of multi-criteria recommender systems. *World Wide Web*, 10, (2007), 415-441.
29. Matsuo, Y.; Hamasaki, M.; Mori, J.; Takeda, H.; and Hasida, K. Ontological consideration on human relationship vocabulary for FOAF. *Proceedings of the 1st Workshop on Friend of a Friend, Social Networking and Semantic Web*. 2004.
30. McGinty, L. and Smyth, B. Adaptive selection: an analysis of critiquing and preference-based feedback in conversational recommender systems. *International Journal of Electronic Commerce*, 11, 2 (2006), 35-57.

31. Nikolaeva, R. and Sriram, S. The moderating role of consumer and product characteristics on the value of customized on-line recommendations. *International Journal of Electronic Commerce*, 11, 2 (2007), 101-123.
32. Palmisano, C.; Tuzhilin, A.; and Gorgoglione, M. Using context to improve predictive modeling of customers in personalization applications. *IEEE transactions on knowledge and data engineering*, 20, 11 (2008), 1535-1549.
33. Panniello, U.; Tuzhilin, A.; Gorgoglione, M.; Palmisano, C.; and Pedone, A. Experimental comparison of pre- vs. post-filtering approaches in context-aware recommender systems. *Proceedings of the third ACM conference on Recommender systems*. New York: ACM Press, 2009, pp.265-268.
34. Polat, H. and Du, W. Privacy-preserving collaborative filtering. *International Journal of Electronic Commerce*, 9, 4 (2005), 9-35.
35. Resnick, P.; Iacovou, N.; Suchak, M.; Bergstrom, P.; and Riedl, J. GroupLens: an open architecture for collaborative filtering of netnews. *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*. New York: ACM Press, 1994, pp.175-186.
36. Saaty, T.L. Decision making with the analytic hierarchy process. *International Journal of Services Sciences*, 1, 1 (2008), 83 - 98.
37. Sampson, S.E. and Froehle, C.M. Foundations and implications of a proposed unified services theory. *Production and Operations Management*, 15, 2 (2006), 329-343.
38. Sarwar, B.; Karypis, G.; Konstan, J.; and Reidl, J. Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th international conference on World Wide Web*. New York: ACM Press, 2001, pp.285 - 295.
39. Schafer, J.B.; Konstan, J.A.; and Riedl, J. E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5, (2001), 115-153.
40. Schilit, B.; Adams, N.; and Want, R. Context-aware computing applications. *1st International Workshop on Mobile Computing Systems and Applications*. New York: ACM Press, 1994, pp. 85-90.
41. Shani, G. and Gunawardana, A. Evaluating recommendation systems. *Recommender Systems Handbook*, New York: Springer, 2011, pp.257-297.
42. Shardanand, U. and Maes, P. Social information filtering: algorithms for automating "word of mouth". *Proceedings of the SIGCHI conference on Human factors in computing systems*. New York: ACM Press, 1995, pp.210-217.
43. Stewart, K.J. and Malaga, R.A. Contrast and assimilation effects on consumers' trust in internet companies. *International Journal of Electronic Commerce*, 13, 3 (2009), 71-93.
44. Symeonidis, P.; Nanopoulos, A.; Papadopoulos, A.N.; and Manolopoulos, Y. Collaborative recommender systems: combining effectiveness and efficiency. *Expert Systems with Applications*, 34, 4 (2008), 2995-3013.
45.Wille, R. Formal concept analysis as mathematical theory of concepts and concept hierarchies. In B. Ganter, G. Stumme and R. Wille (eds.), *Formal Concept Analysis. Foundations and Applications*. Berlin: Springer, 2005, pp. 1-33.
46. Zhang, Y.; Zhuang, Y.; Wu, J.; and Zhang, L. Applying probabilistic latent semantic analysis to multi-criteria recommender system. *AI Communications*, 22, 2 (2009), 97-107.

Figures and Tables

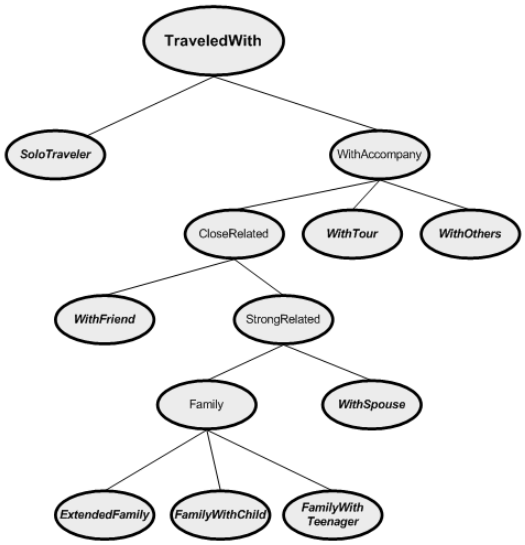


Figure 1. TravelledWith Context Hierarchy Structure

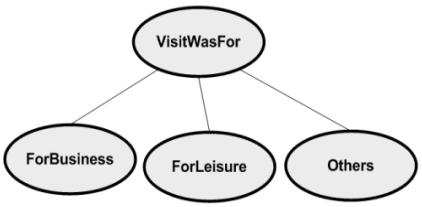


Figure 2. VisitWasFor context structure

Fig. 1 is mentioned in Page 16. It presents the hierarchy structure of TravelledWith context in this paper. Fig. 2 is mentioned in Page 17. It shows VisitWasFor context hierarchy structure.

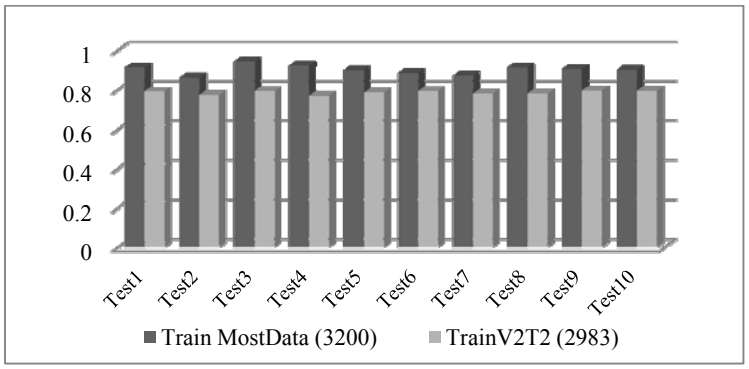


Figure 3. MAEs for testing the impact of context

Table 1. T-Test from Training $V2T2$ and Training set *MostData*

	TrainV2T2 (2983)	*Train MostData (3200)*
Mean	0.78607	0.9021
Variance	9.06979E-05	0.000580758
Observations	10	10
Hypothesized Mean Difference	0	
df	12	
t Stat	-14.15994676	

P(T<=t) one-tail	3.74928E-09
t Critical one-tail	1.782287548
P(T<=t) two-tail	7.49856E-09
t Critical two-tail	2.178812827

Table 1 and Fig. 3 are the results of the first experiment, which is testing whether there is a positive impact of context in predictions. And both of them are discussed in Page 29.

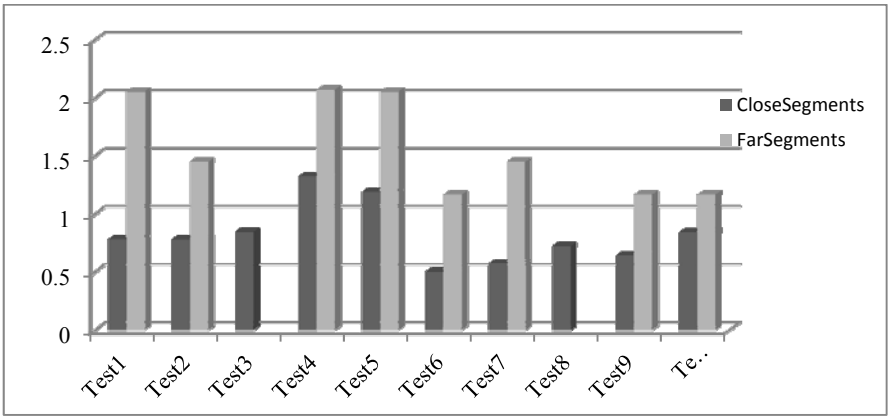


Figure 4. MAEs for testing whether the inclusion of similar context segments produces more accurate prediction

Table 2. T-Test from Training CloseSegments and Training FarSegments

	CloseSegments	*FarSegments*
Mean	0.739423	1.392064
Variance	0.042397	0.229461
Observations	10	8
Hypothesized Mean Difference	0	
df	9	
t Stat	-3.5969	
P(T<=t) one-tail	0.002888	
t Critical one-tail	1.383029	
P(T<=t) two-tail	0.005776	
t Critical two-tail	1.833113	

Table 2 and Fig. 4 are the results of the second set of experiments, which test whether closer context segments matter in prediction. Table 2 and Fig. 4 are mentioned in Page 31.

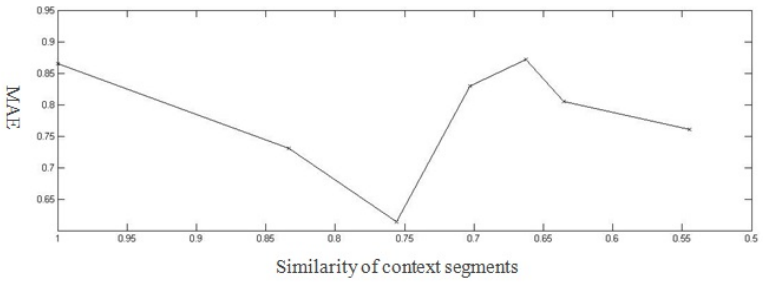


Figure 5. Relationship between quantity of segments and accuracy

Fig. 5 is mentioned in Page 33. It shows the changes of prediction accuracy by increasing the quantity of "borrowed" similar context segments.

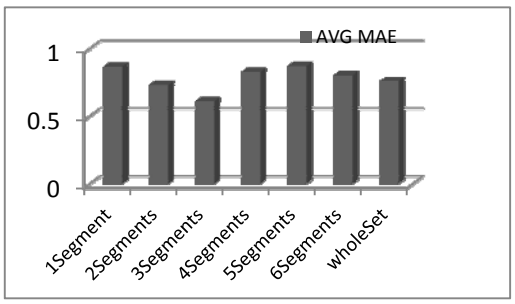


Figure 6. Average MAE with increasing segments

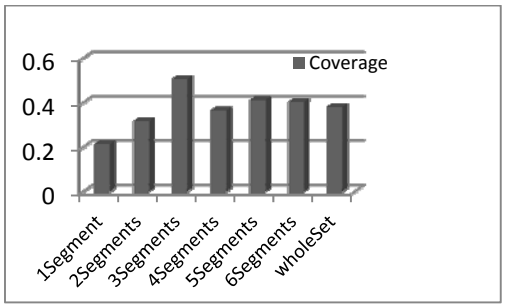


Figure 7. Average coverage with increasing segments

Fig. 6 and Fig. 7 demonstrate the changes of the prediction accuracy and coverage following adding up segment one by one from closest to the final whole set with the limit of computation power. They are first mentioned in Page 34.

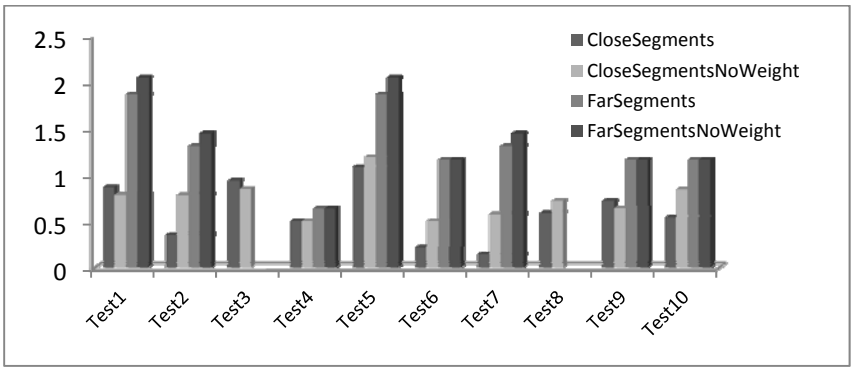


Figure 8. MAEs for testing whether context weight matters

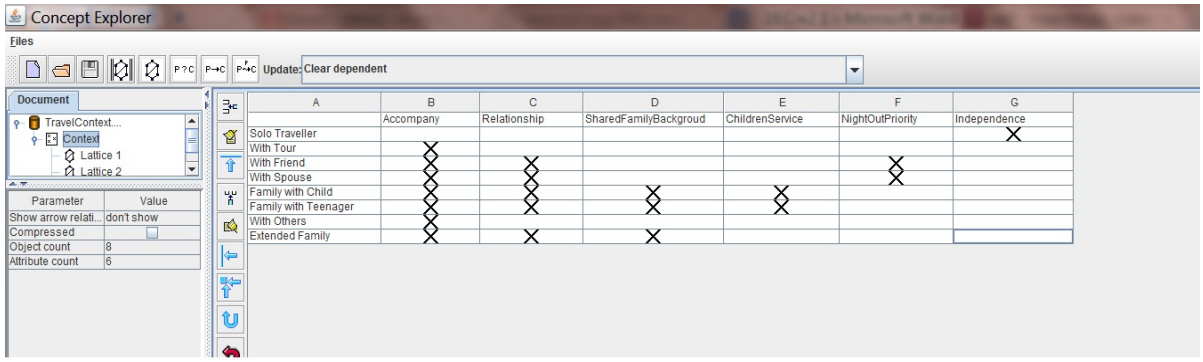
Fig. 8 is discussed in Page 35, as the results of the fourth set of experiment, which tests whether considering context weight in prediction can improve the prediction accuracy or not.

Appendix

The table below shows the average of all the results from our questionnaires.

	Solo	Spouse	Friend	WithChild	With-Teenager	Extended	Tour	Others
Solo	9	5.75	6.25	3.75	3.75	4.25	5.375	4.428571
Spouse	5.75	9	4.875	6	6.125	5.375	4.571429	4.142857
Friend	6.25	4.875	9	4.25	4	4.5	6.75	5.857143
WithChild	3.75	6	4.25	9	7.625	6.875	4	4.857143
With-Teenager	3.75	6.125	4	7.625	9	6.6	3.6	3.5
Extended-Family	4.25	5.375	4.5	6.875	6.6	9	5	5.857143
Tour	5.375	4.571429	6.75	4	3.6	5	9	5.5
Others	4.428571	4.142857	5.857143	4.857143	3.5	5.857143	5.5	9

The figure below is the screenshot of clustering context through FCA technique in Concept Explorer.



Authors

Liwei Liu (liwei.liu@mbs.ac.uk), is a research associate at Manchester Business School, the University of Manchester, UK. She received her Ph.D. in Information System from the same university. Her research interests include recommender systems, context-aware computing, multiple criteria decision making, web services, ontologies, and semantic reasoning.

Postal Address: *Room 3.17, MBSW, Booth Street West, Manchester Business School, the University of Manchester, Manchester, UK, M15 6PB*

Nikolay Mehandjiev (n.mehandjiev@manchester.ac.uk), is Professor of Enterprise Information Systems at Manchester Business School, the University of Manchester. He has published over 100 peer reviewed papers, has co-authored two books and has guest edited four special issues of international journals. He researches the design of flexible service systems, focusing on intelligent service selection and composition models.

Postal Address: *Room 3.24, MBSW, Booth Street West, Manchester Business School, the University of Manchester, Manchester, UK, M15 6PB*

Dong-Ling Xu (L.Xu@mbs.ac.uk**)** is a professor in the Decision and Cognitive Sciences Research Centre, Manchester Business School, the University of Manchester. She has published over 150 papers with many in peer reviewed highly regarded journals. Her current research interests are in the areas of decision making and decision support under uncertainty, and their applications in consumer preference identification, supplier selection, environmental impact assessment, and sustainability management. She developed several statistical pipeline leak detection systems for companies such as CNOOC, Shell and BP. She has also developed several decision support tools including IDS (Intelligent Decision System) which is used by researchers and practitioners from over 50 countries.

Postal Address*: F37 MBSE, Booth Street East, Manchester Business School, The University of Manchester, Manchester, UK, M15 6PB*