A Decision Tree Approach to Voice-enable Mobile Commerce Applications

Yandong Fan[1], Elizabeth A. Kendall[2]

[1] School of Network Computing, Faculty of IT, Monash University, Australia

[2] School of Network Computing, Faculty of IT, Monash University, Australia

Abstract. Speech interfaces have become increasingly popular to support mobile commerce. Although spoken dialogue systems have been studied for decades, they pertain to specific domains. There is a lack of research on general approaches for voice enabling. In this paper, we propose a decision tree approach to personalize and voice-enable applications in the context of mobile commerce. The system dynamically analyses user preferences by mining the navigation and transaction logs. The user profile then is used to personalize a tree-based product catalogue. We utilize the Predictive Model Markup Language to store the resultant catalogue. This XML document is used to construct conversational dialogues for users to find the product information. The proposed architecture has been verified and evaluated though implementing a mobile car city application.

1 Introduction

Mobile commerce (m-commerce) has attracted much interest among researchers, developers and service providers [16]. While anticipating the brilliant potential, currently we are still confronting significant challenges in providing convenient user interfaces for m-commerce. Speech interfaces have become increasingly popular to support information retrieval and online transactions in the mobile environment, as they are the most natural and convenient channels for human communications [8].

Speech-enabled applications for information retrieval in specific domains have been introduced for many years, but application developers still struggle in building new applications due to a lack of general approaches for voice-enabling [8]. More importantly, previous dialogue systems normally have a fixed conversational structure for each user, providing little to no personalization to adapt with consumers' specific interests. With the significant increase of information and services available to customers within a business, it is crucial to the success of an application to personalize the dialogue system according to an individual user's preferences.

In this paper, we propose a decision tree approach combining personalization techniques to voice-enable m-commerce applications. The approach uses a modular architecture. It dynamically analyses user preferences on different products by mining the navigation and transaction logs. Therefore user profiles can be built unobtrusively. The user profile then is used to personalize a tree-based product catalogue. We utilize the Predictive Model Markup Language (PMML 2.0) [3] as the standard to store the resultant catalogue, and this XML document acts as the source to construct conversational dialogues that help users find the product information they need.

The paper is structured as follows. The second Section introduces research on spoken dialogue systems. The third Section presents the overview and the detailed description of the proposed architecture. Implementation and evaluation of the architecture are introduced in the fourth Section. In the fifth Section, related work is discussed. Finally, the sixth Section presents conclusions and discussion.

2 Speech Interface

Research on how to build spoken dialogue systems has been conducted for decades. Initially, most of these systems adopted template-based approaches, which maintain a conversation with a user through applying a set of static and predefined rules [8]. The MERCURY system [11, 12], providing information about flights available for over 500 cities worldwide, has been regarded as one of the most sophisticated systems to date in terms of its dialogue model [12]. However, it is still designed to be specific to a particular domain, where the information needed to locate a flight can be limited to departure point, destination and date of travel. Command-based dialogue systems [7,9] also have been introduced for specific purposes, such as online news browsing for visually impaired people, and mobile communications for car drivers. However, most of these systems maintain a small vocabulary of commands. While moving to m-commerce applications, the situation is different. Accurately locating a product from a catalogue to meet a customer's interest and preferences is more difficult. Features used to describe a product class might consist of general data (price), aesthetic information (size, shape, colour), technical merits (functionalities, advantages) and other factors. Furthermore, in the context of m-commerce, users are often in time-critical situations and their needs are less certain, which requires the system to help and guide users to find the product information they need in a timely manner. Consequently, a personalized dialogue pertaining to each individual customer is required.

Pargellis et al. [8] introduce an automatic dialogue generation platform for personalized dialogue applications. This platform compensates one of the main drawbacks of conventional dialogue systems — creating predefined and static speech dialogues without considering personalization. However, the system required users to maintain profiles for themselves by filling in a set of Web-based pages. Datta et al. [2] argue that such static profiling mechanisms might be difficult to implement, as users are unwilling to update those web pages regularly. Instead, they believe dynamic profiling techniques can outperform their static counterparts.

In this paper, we propose a dynamic profiling approach to unobtrusively predict users' preferences on different product classes in the catalogue. Such user profiles then are used to personalize the tree-based product catalogue for customers.

3 System Descriptions

The architecture we proposed provides a voice interface for human-computer interaction that uses a dialogue structure personalized to each individual user. The customized dialogue structure is created based on a personalized decision-tree catalogue. The architecture consists of five modular components, as is shown in Figure 1 (web and application server).

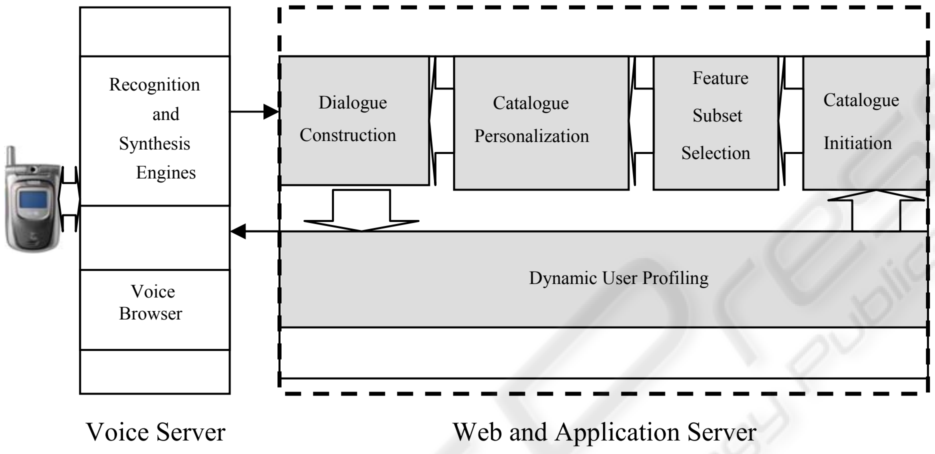

Fig. 1. An architecture for voice-enabling m-commerce applications

The Dialogue Construction Module (DCM) generates customized dialogue structure for a customer to browse product information. By tracing the user's behavior of online voice-browsing and shopping, the dialogue manager within the DCM stores the navigation and transaction records in log files. These log files are mined by the Dynamic User Profiling Module (DUPM) to create a dynamic user profile for each customer. The profile consists of a set of numeric popularity values for products to indicate the customer's preferences and interests. The Catalogue Initiation Module (CIM) uses an original set of features (attributes) to describe product classes in the catalogue, including the numeric popularity values. In order to facilitate feature subset selection and personalized tree induction in the following stages, the CIM is also responsible for discretizing numeric variables into categorical variables. The goal of the Feature Subset Selection Module (FSSM) is to eliminate the existence of irrelevant or redundant attributes. Based on the resultant set of attributes from the FSSM, tree catalogues pertaining to each individual customer's preferences are constructed in the Catalogue Personalization Module (CPM). Dialogues are continuously generated in the DCM to help customers traverse the personalized tree catalogue and find desired products.

3.1 Dynamic User Profiling

The concept of dynamic user profiling is the idea of storing as much historical customer session data as possible, and then mining the navigation and transaction logs to discover usage patterns and other useful information [2]. In our system, the main objective of the DUPM is to unobtrusively determine the popularity of a product in the catalogue in relation to a particular customer. We use a similar approach as Cho et al.'s [1] to construct a product popularity model. Considering a user's shopping experience as a state transition model, there are five possible states that a customer might be in within a session:

1. voice-browsing product information;
2. locating a product and listening to the description of the product;
3. selecting the product for the shopping cart;
4. purchasing the product;
5. terminating the session.

A customer might exit the system at any of states 1, 2, 3 or 4. Through tracking all sessions that have happened with each customer, we can record all occurrences of locating, selecting and purchasing each product in the catalogue. In order to compare the popularity of different products, it is reasonable to assign different weights to occurrences of locating, selecting and purchasing in an ascendant order. Thus we obtain a compound occurrence number C_{ij}:

$$C_{ij} = P_{ij} *a + S_{ij} *b + L_{ij} *c \qquad \textbf{(1)}$$

where P_{ij} is the number of occurrences of purchases of customer i on product class j, and S_{ij} and L_{ij} are the numbers of occurrences of selecting for shopping cart and listening, respectively. a, b, c are weights, and a>b>c.

Thus, each product has a total number of weighted occurrences of locating, selecting and purchasing for each customer. In order to truly personalize the catalogue, we then normalize the results to obtain a popularity value of each product for customer i, which represents this customer's interest in each product class relative to other classes.

The dynamic profiling is periodically updated when new records have been accumulated to reflect the change of preferences and interests of customers. Therefore, instead of explicitly asking customers to specify certain interests or preferences, we unobtrusively collect the voice-browsing and shopping behaviours of customers to determine the popularity values of products, which implicitly reflect their potential interest and buying in the future.

3.2 Catalogue Initiation

Each product class differentiates itself from others by attribute-value pairs. For example, a car model might distinguish itself from others by manufacturer, body style, number of doors, drive wheels, number of cylinders, colour and price. The DUPM adds an extra dimension to differentiate a product from others – popularity value.

The task for the CIM is to select an appropriate attribute-value matrix (AVM) to model the product information. If the task is to voice-enable existing e-commerce applications, the attribute-value pairs can be obtained from existing database; otherwise they may be built from scratch. This indicates that the original attribute set chosen by the designer might not be the optimizing solution for classification, which suggests the necessity of feature subset selection. As decision trees [1, 14] have been selected in our architecture for catalogue presentation, some pre-processing tasks have to be done in the CIM.

A decision tree is a classification model consisting of a set of samples that can be characterized by the same set of attributes (independent variables) and a known class label (dependent variable). For the catalogue tree in our system, we select the popularity value as the dependent variable in the classification model. We have to, however, convert the numeric values of popularity into discrete values because decision tree induction algorithms can only handle categorical variables. Such conversions have to be conducted for other independent variables with continuous values, such as price. These processes are done by the CIM. We choose a relatively simple solution to perform the discretization, which is called *equal-width-intervals*. By identifying the minimum and maximum values for a given numeric variable, we generally divide the range into equal-width intervals.

The selection of popularity as the dependent variable helps with personalization in the next stage. More importantly, it can also be used to predict the popularity class for a new product class. Any new product class, for which there are no navigation and transaction records, can still be assigned a popularity class based on its known attribute-value pairs. In addition, grouping popular products increases the opportunity of cross selling.

3.3 Feature Subset Selection

Both theoretical analyses and experimental studies indicate that many classification algorithms perform poorly due to the existence of irrelevant or redundant attributes in the sample set [5]. Thus, feature subset selection became an important procedure for data pre-processing. The objective of feature subset selection is to find a minimum subset of features that still can satisfy a predefined criterion. Feature selection algorithm can be classified as a filter or a wrapper [5]. A filter is independent of the learning algorithm while a wrapper is intertwined with the learning task. In our architecture, the criterion for feature subset selection is different from the metric used for decision tree induction. Therefore, the FSSM has been implemented as a filter. The three principal dimensions of feature selection are search strategy, evaluation measure and feature generation scheme [5]. Numerous algorithms have been introduced in the literature. We utilize the ABB algorithm [6] for the FSSM. The algorithm uses inconsistency rate for feature evaluation. The inconsistency rate of a data set is defined as the following:

- Two instances are inconsistent if they match on all features but mismatch on the class label;
- For a set of matching instances, the inconsistency count is the number of instances in the set minus the number of instances belonging to a major class in the set;

- The inconsistency rate is the sum of all inconsistency counts divided by the total number of instances.

The ABB algorithm tries to find a minimum size of feature sets that separate classes as consistently as the full set of features. Therefore we choose the minimum subset of features that can keep the consistency of the samples maintained by the full set of features. Both irrelevant and redundant features can be removed by using consistency measures [5].

3.4 Catalogue Personalization

A good catalogue for speech-enabled applications should help consumers to find products of interest in a timely manner. Although traditional decision tree induction algorithms such as C4.5 [10] can provide a general approach for catalogue construction, they are not specifically designed to personalize speech applications. Firstly, the selection procedure of attributes in traditional induction algorithms relies on metrics such as gain ratio, which have nothing to do with personalization. Secondly, the computation of such metrics is complicated, which leads to a significant delay for tree completion. Finally, the tree structure constructed by traditional algorithms does not consider the constraint of speech interfaces — a person can only remember less than 7 choices when confronting selection [13].

We introduced a new algorithm to construct the tree-based catalogue. The objective of the algorithm is to minimize the average depth of the product hierarchy by moving the popular product classes to the upper level of the decision tree so that consumers can find them quickly. To standardize the meaning of the average depth of a tree, we proposed a new metric to evaluate the extent of personalization, which we called the *personalization degree*. It is defined as:

$$Pd = 1/[\Sigma(\mathrm{Pop}_j * n_j)], \qquad j=1\ldots N \tag{2}$$

where Pop_j is the numeric popularity value of product class j, and n_j is the level value of product class j in the tree.

The algorithm can be divided into three steps. Firstly, it constructs deferent trees based on the selection sequence of attributes. Secondly, based on the fact that a person can easily forget what has been prompted by the system, sorting and division have been conducted to group product classes by five in each leaf node if the number of classes is greater than five. Finally, as all the possible tree structures have been normalized from the previous steps, the algorithm computes the personalization degree for each tree structure, and then selects one with a maximum value as the resulting tree.

In order to increase the generality of applications development, we adopted PMML [3] as the XML vocabulary to store the resulting tree catalogue. This makes the transformation between the product information (in PMML) and the speech presentation (in VoiceXML) easy. In addition, new product classes can be inserted into the PMML model through predicting their popularity classes.

3.5 Dialogue Construction

As the PMML catalogue has been personalized to an individual customer, the corresponding speech presentation is also pertaining to a customer, which means that each customer may has a different user experience from others. It is possible to create a static structure of dialogues that can match with the PMML-based tree catalogue. But that means we have to maintain one set of dialogues for each customer permanently in the Web server, and most of the dialogues would not be visited by a customer within a session as he or she might select a certain path to traverse the nodes in the tree. It is unnecessary and not economical to implement such static approaches. Alternatively, we should consider implementing a dialogue generator engine that can dynamically create dialogues for customers based on their interaction with the system.

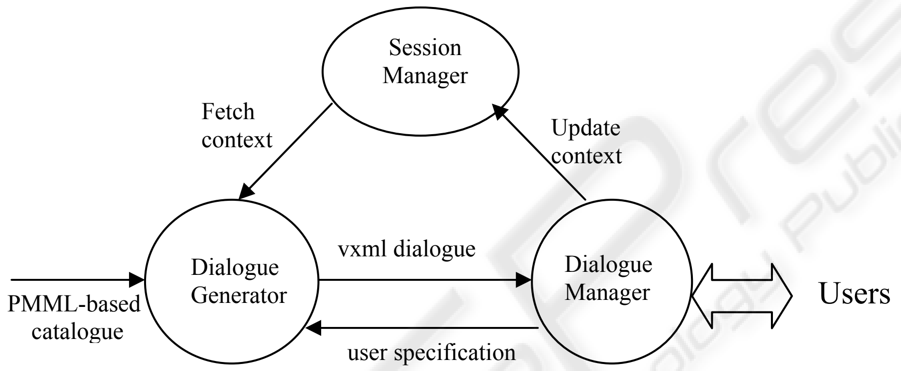

Fig. 2. The dialogue construction model

Figure 2 represents the functional components of the DCM. A dialogue session between the system and a user is an actual conversation that consists of a specific series of dialogue turns. While the user traverses the tree nodes to find out products of interest, it is the Session Manager's responsibility to remember the current context the user is working with. Since the goal is to maintain a dynamic dialogue, the Dialogue Generator has to constantly trace the context of the previous dialogues in order to generate the next dialogue for further interactions.

The Dialogue Manager continuously manages dialogue turns one by one. This looping process continues until the user terminates the session. The Dialogue Manager receives the user's utterance and recognizes its semantic meanings. This user specification then is passed to the Dialogue Generator. In order to generate the next dialogue for the Dialogue Manager, the Dialogue Generator has to know three main pieces of knowledge: the history of previously visited nodes in the tree, the current node specified by the user, and the list of associated values of the current attribute node. The first piece of knowledge comes from the Session Manager. The second piece of knowledge is indicated by the user specification. The third one is obtained from parsing the PMML tree-based catalogue. In our system, we select VoiceXML for dialogue management testing. In real applications, the conversational system can be implemented by proprietary speech APIs that are specific to a particular platform.

4 Implementation and Evaluation

We verify and evaluate the proposed architecture through implementing a mobile car city application, which allows mobile phone users to browse car model information and proceed to book an appointment for viewing the desired model. The original product catalogue is adapted from a data set introduced by the UCI Repository [15]. The catalogue consists of 312 instances of car models, where each car model is described by eleven attributes: make (MK), number_of_door (ND), body_style (BS), price (PR), number_of_cylinders (NC), drive_wheels (DW), color (CR), year_made(YM), number_of_passengers(NP), engine_litre(EL), and transmission_type(TT). Through simulating user interactions with the system, we obtain the necessary navigation and transaction records and compute popularity values of products for each customer. As well, numeric values have been discretized into categorical values.

We then apply the ABB algorithm to select a subset of features for tree induction. Results show that the only subset that can satisfy our criterion is {MK,BS,DW,CR,YM,EL,TT}. The subset selection significantly reduces the search space for a best tree. We then apply the tree personalization algorithm to this subset of features. The resulting tree has a personalization degree of .216. Figure 3 shows a fraction of the resultant tree-catalogue in PMML. Based on the structure of the PMML document, we construct application-directed dialogues to guide customers to find desired products. The dialogue manager is implemented in VoiceXML. Users can use any VoiceXML-compatible voice browser to access the application.

```
<?xml version="1.0"?>
<PMML version="2.1">
  <TreeModel modelName="CarModels" function="classification">
    <Node id="0" name="body_style" score="L" recordCount="63">
      <True />
      <Node id="1" name="color" score="L" recordCount="26">
         <simplePredicate field="body_style" operator="equal"
           value="hatchback" />
         <Node is="2" name="engire_litre" score="L" recordCount="6">
          <SimplePredicate field="color" operator="equal" value="red" />
          <Node id="3" name="make" score="L" recordCount="3">
           <SimplePredicate field="engine" operator="equal" value="3.0"/>
           <Node id="4" name="leaf" score="M" recordCount="2">
            <SimplePredicate field="make" operator="equal" value="Honda"/>
            <Extension>
              <Array n="2" type="String">1 8</Array>
            </Extension>
           </Node>
            ……
  </TreeModel>
</PMML>
```

Fig. 3. PMML tree-based catalogue

The evaluation is performed in two levels. Firstly, the decision tree model is evaluated by its classification and predictive accuracy. We first use the entire data set for training. The resultant tree has a classification accuracy rate of 92.3%. Then the data set is split randomly into two sets, 208 instances for training and 104 for testing. The predictive accuracy is 76%. The performance is compatible with results reported in [4]. Secondly, two colleagues help to test the acceptance of the speech interface.

The result is acceptable in terms of word error rate (WER) of voice recognition (less than 7%).

5 Related Works

The pioneering work on the application of decision trees to catalogue construction was reported in [14, 18], where decision trees were used to represent online catalogue topologies. Each leaf node is an individual product page. Our approach is different from their work in two aspects. One is that the decision tree model in our approach has predictive capability by using popularity as class label. The other is that our model is specifically designed for voice-enabled applications by considering limitations of speech interface.

[8] introduces an automatic dialogue generation platform. It shares the same objective with our approach to provide a VoiceXML-compatible speech interface personalized to an individual user. Compared with [8], our approach utilizes dynamic profiling instead of static ranking, and our decision tree model is simpler and more effective in voice recognition. Researchers in MIT [12, 17] explore general approaches to develop mixed-initiative spoken dialogue systems for telephony services. We prefer an application-directed dialogue structure in our approach, which is more suitable in the context of mobile commerce.

6 Conclusions and Discussion

The proposed architecture provides a general approach for voice-enabling of m-commerce applications. To our knowledge, our study is the first of its kind to incorporate dynamic personalization techniques into decision-tree based catalogues for speech-enabled applications. [16] identifies 12 important classes of m-commerce applications. We believe that typical m-commerce applications such as mobile shops and music-on-demand, in which product information can be described as attribute-value matrix, can be voice-enabled and personalized using the proposed architecture. In general, the architecture has several features. Firstly, the customer preference model has been established implicitly through mining navigation and transaction logs. Thus it avoids intrusively requiring for user ranking on product classes. Secondly, new products, for which there are no navigation and transaction records for mining, can be inserted into the PMML catalogue through prediction of popularity class. Finally, with personal catalogues, products with higher popularity values can be more easily found than those with lower popularity values, which is crucial for users in m-commerce context.

There are also some concerns associated with the architecture. Firstly, an obvious question might be how we can identify a customer in order to render personalized content for him or her. As we are dealing with cellular phones, each mobile phone has a unique identified number. This number can be fetched from the telephony network to the Internet. In the VoiceXML dialogue manager, this information is stored in the *session.telephone.ani* variable. We can use it to identify the caller. If the caller's number is not available, then a login session might be implemented to identify who the user is. Secondly, the product popularity model in our system computes the popularity value for each product class according to the navigation and transaction logs. This assumption cannot apply to new customers or customers who use the application sparsely. This limitation is not unique to our approach. Under such circumstances, a possible solution might be clustering customers into groups based on similarities. Thus the navigation and transaction logs from a group of customers can be used to calculate popularity values for products.

References

1. Cho, Y.H. et al. A personalized recommender system based on web usage mining and decision tree induction. *Expert Systems with Applications, Vol.23* (2002), pp.329-342.
2. Datta, A. et al. An architecture to support scalable online personalization on the web. *The VLDB Journal, vol.10* (2001), pp.104-117.
3. Data Mining Group: PMML 2.1. Retrieved May 25, 2004, From: http://www.dmg.org
4. Liu, H. and Setiono, R. Feature Selection via Discretization. IEEE Transactions on Knowledge and Data Engineering. Vol.9, No.4, Jul/Aug 1997, pp.642-645.
5. Liu, H. and Motoda, H. *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, 1998.
6. Liu, H. et al. A monotonic measure for optimal feature selection. *Machine Learning: ECML-98*, Springer-Verlag, 1998.
7. Minker et al. The SENECA spoken language dialogue system. *Speech Communication*, Article in Press, 2004.
8. Pargellis, A. N. et al. An automatic dialogue generation platform for personalized dialogue applications. *Speech Communication*, *Vol.42* (2004), pp.329-351.
9. Pavešić, N. et al. Homer II — man-machine interface to internet for blind and visually impaired people. *Computer Communications, Vol.26* (2003), pp.438-443.
10. Quinlan, J.R. *C4.5: programs for machine learning*. San Mateo, CA, U.S.A., Morgan Kaufman, 1993.
11. Seneff, S. et al. Galaxy-II: a reference architecture for conversational system development. *In Proceedings of the ICSLP'98*, Sydney, Australia, 1998. pp.931-934.
12. Seneff, S. Response planning and generation in the MERCURY flight reservation system. *Computer Speech and Language, Vol.16 (*2002), pp.283-312
13. Sharma, C. and Kunins, J. VoiceXML: Strategies and Techniques for Effective Voice Application Development with VoiceXML 2.0. Wiley Computer Publishing, 2002.
14. Sung, W.-K. et al. Automatic Construction of Online Catalog Topologies. *IEEE Transaction on Systems, Man, and Cybernetics — Part C: Applications and Reviews, Vol.32* (2002), No.4, pp.382-391.
15. UCI Machine Learning Repository. Retrieved May 12, 2004, From: http://www.uci.edu
16. Varshney, U. and Vetter, R. A framework for the emerging mobile commerce applications, *Proceedings of the 34th HICSS*, 2001, pp.1-10.
17. Weinstein, E. SpeechBuilder: Facilitating Spoken Dialogue System Development, Master Thesis, MIT, 2001.
18. Yang, D. et al. Construction of Online Catalog Topologies Using Decision Trees. *Proc.2nd Int'l Workshop Advance Issues of E-Commerce and Web-based Information Systems (WECWIS 2000)*, IEEE CS Press, Los Alamitos, CA, 2000. pp.223-230.