Predictive Menu Selection on a Mobile Phone

Robert Bridle and Eric McCreath

Department of Computer Science, Faculty of Engineering and Information Technology, The Australian National University

Abstract. A mobile phone is often characterised by its limited user interface. We consider applying a machine learning approach to improve the usability of a mobile phone's interface. We present an approach that predicts a user's menu selection on a mobile phone. The learning approach consists of a highly restricted set of hypotheses. A restricted set of hypotheses allows both the learner to operate efficiently within the limited hardware resources of a mobile phone and learn concepts quickly from a small training set. We compare our approach with alternative menu prediction approaches in a simulation. Also a modified Nokia$^{\text{TM}}$Series 60 mobile phone address book application was developed to incorporate our menu prediction approach. A user evaluation of this implementation shows that our predictive menu selection approach has the potential to reduce the number of key presses a user makes.

1 Introduction

The mobile phone has become a ubiquitous device, providing mobile access to information and communication services. Yet due to the physical limitations of the interfaces on these devices they can be cumbersome to use. A current mobile phone such as the Nokia$^{\text{TM}}$6220, has a colour screen with 128x128 pixels and 25 buttons which are generally pressed with your thumbs. Considering these restrictions a mobile phone's interface must be carefully designed in terms of useability. The aspects important in mobile phone interface design are: efficiency and predictability[14]. Users generally like to direct the device to carry out their tasks with as few button presses as possible. Also, users like to be able to predict what will happen when options are selected. To handle the balance between efficiency and predictability we focus on applying a learning approach to the menu elements of a mobile phone interface. When a user is presented with a menu, the learning system will predict a menu-item and highlight it. Since a mobile phone interface is primarily menu-driven this approach will reduce the number of scrolling key presses the user has to make. Furthermore, the predictability of the menu is retained since the ordering of menu-items does not change.

A mobile phone provides two more challenges for a learning system. Firstly, mobile phone devices have limited memory and processing power. So an extensive search of a large hypothesis space would not be practical. Moreover, this would also require an hypothesis to be represented and interpreted as the user interacts with the system. Any approach must collect data and make predictions in an "on-line" and "real-time" manner. Secondly, users will often change the task they are performing on a mobile phone. For example a user may be adding new contacts to their address book, which can be captured by the concept of adding new contacts and then assigning them to groups. They then may need to edit some contacts, the concept changes to opening and editing contacts as a result. The learning system must be able to react to these concept changes with only a very small amount of data. To meet these requirements we provide the learner with a greatly restricted and tailored hypothesis space. This strong bias enables the learner to find a consistent hypothesis from far fewer training examples.

In the next section we describe related research. In Section 3 we describe the learning framework and algorithms considered. In Section 4 we describe menu selection prediction in an address book application. In Section 5 we compare alternative menu selection approaches in this application. Finally, we evaluate alternative menu selection approaches using real user data collected from this application.

2 Related Work

Predicting menu selections can be defined as an adaptive user interface problem. A considerable amount of research as been conducted into adaptive user interfaces. Our menu prediction approach attempts to learn a concept that can be used to predict a user's next action, hence it can be considered a generative adaptive interface[12]. Many generative interface applications exist, these include; programming by demonstration[2], scheduling[16] and user modelling applications[5, 13]. The generative interface task that most closely matches ours is that of command line prediction[10, 3, 4]. This research shares many of the challenges we face in applying machine learning to menu interface prediction. Both must predict the next action a user is going to perform by inferring the current task a user is performing, and both must increase interface efficiency while maintaining interface predictability. However, menu selection prediction may be a more assailable task, as the number of possible actions that can be presented in a menu is usually quite small. Also, our approach differs from the majority of command line prediction approaches by not using a probabilistic model to predict a user's next action.

A similar attempt to increase the usability of a mobile computing device was the Names++ application by Schlimmer[20]. This system was built upon previous work performed by Schlimmer and Hermens[8]. Three intelligent user interfaces were evaluated in a address book application on an Apple Netwon$^{\mathrm{TM}}$. It was shown that by using predictive text fill-in or an adaptive menu, the user interface could be up to twice as fast as typing on screen when entering text. Schlimmer's work shares a close similarity with ours in that both investigate adaptive user interfaces on a mobile computing device. Furthermore, we both investigate improving the interface of an address book application. Although similar, our menu selection prediction approach focuses entirely on menu interface elements, while Schlimmer's work considers adapting multiple interface elements.

An early investigation into adaptive menu interfaces was performed by Greenberg and Witten[6], they investigated the effect of re-ordering menu-items according to their frequency of use. The result of their study showed a significant increase in efficiency and a reduction in error-rate when using a frequency-reordered menu. However, later investigations by Mitchell and Shneiderman[15] and by Sears and Shneiderman[21] showed that once users had become familiar with a frequency-reordered menu interface, there was no difference in error-rate or efficiency when compared with statically ordered menus. Moreover, they found that users disliked the unpredictable nature of frequency-reordered menus. Although the results of this research lends support to the notion that user's prefer predictability in a menu, it can not be directly compared with our proposed menu interface. We consider changing which menu-item is highlighted by default, as such our menu can not be consider to be either frequency-reordered or traditionally static.

From a human computer interaction perspective a considerable amount of research has been performed in understanding menu selection through cognitive modeling[1, 17]. However, menu selection differs on a mobile phone in a number ways. Amant, Horton and Ritter[22], describe three features, that make menu selection on a mobile phone unique:

- Selection occurs from key presses (since direct selection capabilities such as a mouse do not usually exist).
- Display sizes are small thus limiting the number of menu-items that can be displayed at once.
- There is less standardisation in menu selection across mobile phones than in desktop machines.

Amant, Horton and Ritter also show that the GOMS model of task analysis closely aligns with real user behaviour on a mobile phone interface. We evaluated our menu selection prediction approach by determining the average number of key presses a user performs, this can be shown to be equivalent to the GOMS model.

A major problem faced by menu selection prediction is that of concept drift. When a user changes the task they are performing, the underlying concept that drives their menu selections also changes, resulting in concept drift. Schlimmer's STAGGER[19] system was an early learning system that addressed concept drift directly. The STAGGER system adjusted to changes in the target concept by triggering revision in a set of concept descriptions. The trigger for revision was based on a measure of logical sufficiency and dependency between the learned concept and new training instances. The task of menu selection prediction is characterised by sudden changes in a target concept, as such we do not consider the STAGGER method. The FLORA[11] learning system and its subsequent versions handled concept drift by selecting relevant instances on which to construct a target concept. The method for determining relevancy was based on selecting training instances bounded by a window into the past. The use of a windowing method can be seen in the original FLORA system and in the PECS[18] system. Adaptive window size methods have also been considered [9, 24]. If a small window size is chosen then it is possible for a learner to handle rapid changes in a target concept. Also, it is desirable in terms of efficiency for a learning system to be able to recognise and reuse recurring target concepts[25, 23]. The approach we describe employs a restricted set of hypotheses and as such relies heavily on reoccurring target concepts existing in the learning environment. We do not consider approaches that address concept drift using contextual clustering. The contextual clustering used by Harries, Sammut, and Horn in the SPLICE[7] system requires training examples to be provided in batch. Our learning task requires concepts to be identified"on-line".

3 Menu Selection Prediction

The aim of the learning system we propose is to predict a user's menu selection on a mobile phone. When a user is presented with a menu, the learning system will predict the menu-item the user will select and highlight it. The intended result of this method is to reduce the number of scrolling key presses made by a user. Other methods do exist to eliminate menu scrolling on mobile phones. One common method uses numbered menu-items, this allows users to jump to a menu-item by selecting the item's corresponding number. This method does not provide an overall solution to the problem since users can only remember a few numbered shortcuts, and once remembered, the mapping of numbers to menu-items cannot change.

To accomplish menu selection prediction, we propose a learning system that will proceed in an "on-line" fashion. As a user interacts with a mobile phone's menu interface they generate a sequence of positive examples. Each example is described by a set of attribute values and a class label. The class label is the menu-item selected by the user and the attribute values are certain properties that the class label may be dependant upon. Every menu interaction performed by a user provides an explicit positive example to the learner (also implicit negative examples are given, e.g. the action the user did not take). After observing these examples the learner's task is to find a concept consistent with the examples. Once the target concept is found, it can be evaluated to determine which menu-item will be selected.

The concepts that may be chosen by the learner are only those concepts that can be represented using the hypothesis space provided to the learner. The hypothesis space we provide to the learner is restricted in a way that only allows likely concepts to be represented. For example, in an address book application we include in the hypothesis space the concept of creating a new contact and then adding that contact to a group, since this is a likely pattern of use. However, we do not include the concept that adds a contact to a group and then deletes that same contact, as a user would generally not interact with the device in this way. In restricting the hypothesis space, and hence the concepts that the learner can choose from, we are focusing the learner on likely concepts and improving the efficiency of the learner in finding a consistent concept. Restricting and tailoring the hypothesis space also improves the ability for the learner to handle concept changes. As a user changes from one task to another, the underlying concept that describes the user's current task will also change. In a mobile phone environment where there are many small tasks being performed by the user, the learner must identify and respond to concept changes quickly if it is to provide relevant predictions. Having a highly restricted and tailored hypothesis space allows the inconsistency of a hypothesis to be identified with only minimal examples from the user. Also, the number of possible concepts that are open for consideration after a concept change occurs is small. Therefore only a few examples are required to identify the new concept.

We now formalize a simple model in which the learning is set. Let C be the class of all concepts. Each concept c_i has a probability associated with it $P(c_i)$ that represents the chance that c_i is selected by the user. We assume the user keeps the same concept from one interaction to another with probability λ_i. Whereas with probability $1-\lambda_i$ we assume the concept is selected from the pool of available concepts. This provides a simple way of modelling concept change.

We are assuming that the concept will change from time to time. Hence, only the data following the most recent change in concept will be of value. Also we assume that there is no noise within the training data. This is not an unreasonable assumption, given the small amount of data we are dealing with. A hypothesis that captures the current concept will be correct on all the previous examples going back to the last change in concept. Hence, the hypothesis going backwards from the current time with the longest correct run is most likely to capture the current concept. This idea forms the basis of the simple algorithm we use. The algorithm works as follows:

- Associate a counter with each hypothesis. Initialize these counters to zero.
- When a new training example $\langle a, l \rangle$ is received each hypothesis h is considered: (a is the set of attributes, l is the menu-item selected)
    - If $h(a) = l$ then increment the counter otherwise reset the counter to 0.
- Use the hypothesis with the highest count to make the prediction.

4 Address Book

We chose an address book application to investigate our learning approach for menu selection prediction. In particular, we chose the *Contacts* address book application found on all Nokia$^{\text{TM}}$Symbian OS$^{\text{TM}}$-based mobile phones. This application was chosen since it is a commonly used function on a mobile phone and is mainly menu-driven. An example of a user interacting with this interface is shown in Figure 1.

The main menu in the *Contacts* address book application consists of 14 menu-items ordered: Open, Call, Create message, New contact, Edit, Delete, Duplicate, Add to group, Belongs to group, Mark/Unmark, Send, Contacts info,

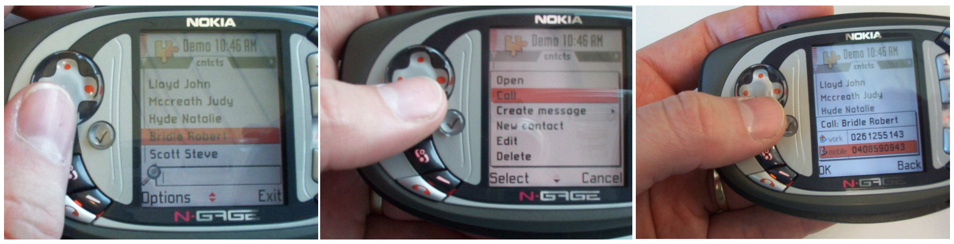

Fig. 1. Using the *Contacts* menu to place a phone call.

Help and Exit. We considered the following 3 attributes that the menu-item may be dependant upon: the last menu-item selected, whether the user has scrolled to a person in the address book and any groups the selected person belongs too.

The hypothesis space that we considered in this learning environment was restricted to a set of decision lists. When evaluated, each decision list predicted one of the 14 menu-items. Each decision list consisted of ordered IF...THEN rules. A rule's condition consisted of any attribute-value test or conjunction of tests using the 3 attributes mentioned above. Table 1 shows part of the hypothesis space (8 relatively simple decision lists). The complete hypothesis space contains 14 repetitions of each of the decision lists shown (112 decision lists in total). Each repetition differs by considering a different menu-item for the default rule (we show those with the default rule **Open**). Also, the X represents a wild card that can take on the value of any group contained in the address book, e.g. Work, Friends, Family. Although many more attributes and concepts could have been provided to the learner we only considered these concepts, since they are most likely given the common tasks users perform in an address book.

We considered the following menu selection prediction approaches:

- First Menu Item – This approach represents a traditional menu which always predicts the first menu-item.
- First Menu Item(frequency re-ordered) – A simple way to improve on the traditional menu approach is to re-order the menu-items in terms of their selection frequency.
- Last Menu Selected – The next menu-item to be selected is the same as the last menu-item selected.
- Most Common Menu Item – This approach predicts the menu-item that has been selected most often using the examples observed so far.
- Most Common Hypothesis – This approach looks at all the previous training data and selects from the restricted hypothesis set the hypothesis most consistent with the examples. This hypothesis is then used to predict the next menu-item to be selected.
- Fixed Window Size – This approach uses a fixed sized moving window to limit the training examples provided to the learner. As a new training example is observed it is added to the front of the window, and the oldest example re-

IF last-menuitem-selected = New Contact **AND** scrolled = true
THEN New Contact
ELSE IF last-menuitem-selected = New Contact **AND** scrolled = false
THEN Add to group
ELSE IF scrolled = false **THEN** New Contact
ELSE (DEFAULT) Open
IF last-menuitem-selected = Open **THEN** Edit
ELSE (DEFAULT) Open
IF last-menuitem-selected = Edit **THEN** Open
ELSE (DEFAULT) Open
IF contact-belongsto-group = X **THEN** Call
ELSE (DEFAULT) Open
IF contact-belongsto-group = X **THEN** SMS
ELSE (DEFAULT) Open
IF contact-belongsto-group = X **THEN** Edit
ELSE (DEFAULT) Open
IF contact-belongsto-group = X **THEN** Delete
ELSE (DEFAULT) Open
(DEFAULT) Open

Table 1. Part of the hypothesis space provided to the learner (8 decision lists). The complete hypothesis space contains each decision list repeated with a different default rule.

moved from the window. The learner is presented with the training examples from the window and selects from the restricted hypothesis set the hypothesis most consistent with the examples. This hypothesis is then used to predict the next menu-item to be selected. We consider window sizes ranging from those contain only 1 example to those that can contain 4 examples.

- Most Recent Correct Hypothesis – This approach represents our learning approach for menu selection prediction, as described in Section 3.

5 A Simulation

To compare our learning approach with the other possible menu prediction approaches we created a simulated dataset. The dataset attempted to model the likely data a user would generate when interacting with an address book application on a mobile phone. We used the *Contacts* address book application described in Section 4 as the basis for the parameters used to create the dataset. In Section 6 we repeat these comparisons on data collected from real users.

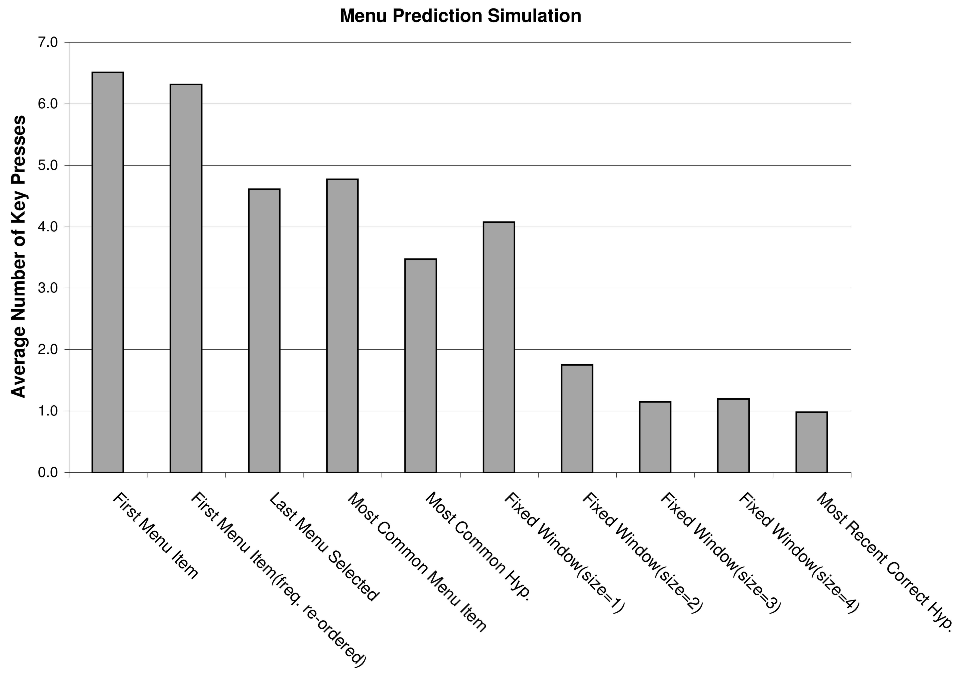

Fig. 2. Simulation results.

We created 112 different concepts[1]. A concept contained each of the 84 possible attribute-value combinations[2] randomly mapped to one of the 14 menu-items. Each of the 112 concepts represented a different random mapping of the attribute-value combinations to a menu-item. To model concept change we generated a sequence of concepts. The sequence was a 100 concepts long. A uniform distribution was used to select the first concept; then for each iteration the current concept was kept with probability $\lambda_i = 0.9$. If the concept changed, a new concept was reselected with respect to the uniform distribution. The sequence was intended to represent a user selecting menu-items according to a concept, with the user changing to a new concept 10 percent of the time.

Each concept in the concept sequence was used to generate a training example. For instance, given the concept: "the user always chooses the *Open* menu-item after they have scrolled otherwise they choose the *Edit* menu-item" remains true for the first 15 elements of the concept sequence. Then the first 15 training examples would represent examples where if scrolling is true then the menu-item attribute equals *Open* otherwise the attribute equals *Edit*. The same sequence of training examples was provided to all of the menu prediction approaches de-

[1] This number is based on the 8 decision lists shown in Table 1, each with 14 different default rules. This is far fewer than most machine learning approaches. (8x14 = 112)

[2] The last menu-item selected attribute could take on 14 values, the scrolled attribute was boolean valued and the groups a contact belonged too could take on 3 values (e.g. Work, Family, Friends). (14x2x3 = 84)

scribed in Section 4. The hypothesis-based menu prediction approaches were given the 112 possible concepts that training exampled were generated from. Each menu prediction approach attempted to predict the next menu-item the user would select. After making a prediction each approach would receive the next training example. This training example would inform the approach on what the user did select. The aim of the simulation was to evaluate how well each menu prediction approach would perform against a dataset that modelled the expected concept-drift characteristics of a real user. Note that we repeated this simulation 100 times and reported the average results. The standard error on the estimates of all these means was below 0.08.

As the aim of menu selection prediction is to minimize the number of scrolling key presses required by a user, we reported the average number of scrolling key presses that would be required given each approach. Since each of the 14 menu-items represents an index in the menu, each menu-item can be represented by an integer, e.g. 0 - Open, 1 - Call, ..., 13 - Exit. The class label is therefore an integer between 0 and 13 (inclusive). The error metric is: $error = |h(a) - l|$. The error metric is the distance between the menu-item predicted and the menu-item desired by the user. The results are shown in Figure 2. The simulation results are encouraging. The results show that our *Most Recent Correct Hypothesis* approach, described in Section 3, performs well when compared with the relatively simple approaches and the other hypothesis-based approaches. Most importantly, the results show that our approach performs well when the training data contains sudden concept changes.

Figure 3 illustrates why the *Most Recent Correct Hypothesis* approach works well when compared with the other hypothesis-based approaches. Given a sequence of training examples 1 through to 4, Figure 3 shows concepts that are consistent with the current example with a tick, and those that are not with a cross. In this example we can see that when deciding on example 5, the hypothesis-based approaches will chose different concepts. The *Most Common Hypothesis* approach will choose the concept that has been correct most often, in this case it will be Concept A. The *Fixed Window* approach with window size 1 will tie between Concept A, B and C. With a window size of 2 Concept C will be correctly chosen. With a window size of 3 a tie between Concept A and C exists and with a window size of 4 Concept A is chosen. Like the *Fixed Window* approach of size 2, the *Most Recent Correct Hypothesis* approach will correctly chose Concept C. However, the *Most Recent Correct Hypothesis* will chose Concept C for example 6 (not shown), while the *Fixed Window* approach of size 2 will tie between Concept B and C. The *Most Recent Correct Hypothesis* approach is not limited by a set window size of examples when evaluating hypotheses.

6 Experimental Results

We implemented a modified version of the *Contacts* address book application on a Nokia N-Gage QD$^{\mathrm{TM}}$mobile phone. The *Contacts* application is an address

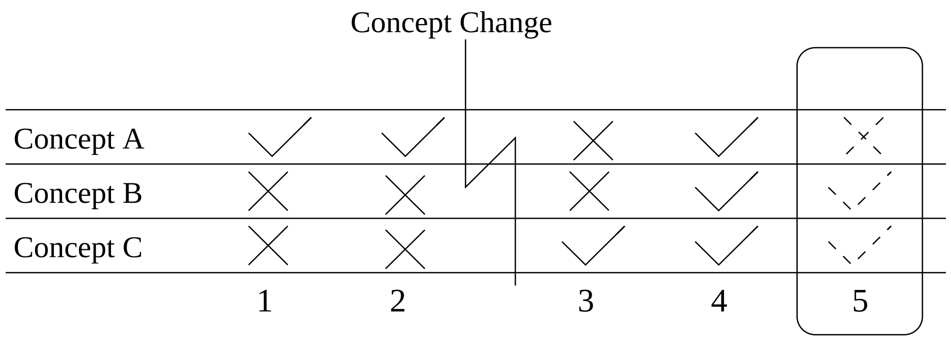

Fig. 3. A sequence of 4 examples are presented to 3 concepts. Ticks represent those examples consistent with the concept, while crosses those that are inconsistent. The concept changes from A to C after example 2.

book application that is installed on all Nokia^{TM}smart-phones. Our version of the *Contacts* address book application recorded the same attributes that we considered in the simulation, see Section 4. We collected training data from two users(the users are the authors) using our implementation of the *Contacts* application. Data was collected over a period of 2 months. The effectiveness of the menu prediction approaches on this data is shown in Figure 4.

First Menu Item: The ordering of menu-items in our version of the *Contacts* application is identical to the ordering used by Nokia^{TM}. Menu items are ordered (Open, Call, Create SMS, New, etc). For both users the *First Menu Item* approach requires substantially less key presses than the expected average of 6.5 key presses, suggesting Nokia^{TM}has chosen a menu ordering that reduces the number of scrolling key presses required by an average user.

First Menu Item(frequency re-ordered): This approach improved on the *First Menu Item* approach. As expected, customising the menu ordering for a particular user will improve the number of scrolling key presses that user needs to make. However, the ordering was derived by inspecting all training instances beforehand. Knowing the selection frequency of menu-items beforehand is an unrealistic assumption.

Last Menu Selected: This approach performed well on the data from the second user, but not on data from the first user. This can be explained by the fact that the second user's data was characterised by long sequences in which they exclusively selected the SMS menu-item. Where as the first user's data did not include long sequences in which they consecutively selected the one menu-item.

Most Common Menu Item: This approach exhibits the same behaviour between the 2 users as the *Last Menu Selected* approach. Again, the second user used the *Contacts* application to predominately send SMS messages, in fact they selected the SMS menu-item 62% of the time. Always predicting the most commonly selected menu-item was advantageous for the second user.

The introduction of hypotheses as shown in the last six approaches in Figure 4, allows the average number of key presses to remain consistently low for

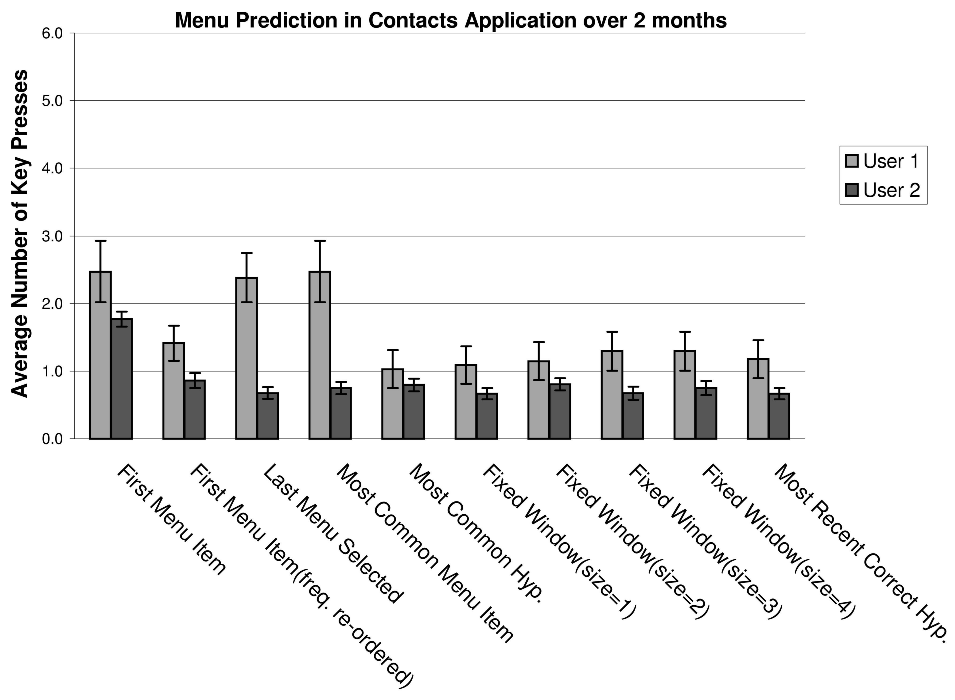

Fig. 4. Menu prediction strategies evaluated on data collected over 2 months from the Nokia^{TM}Contacts application.

both users. The use of a hypothesis space allows the learner to capture concepts exhibited by each user. However, the *Most Recent Correct Hypothesis* approach did not out-perform the other hypothesis-based approaches, instead all the hypothesis-based approaches performed similarly. One reason for this is that evaluating the hypothesis-based approaches using data that has been collected over a long period may not provide an accurate comparison between approaches. Data collected over a long period will contain situations where the one concept remains stable for a period of time, such as adding a new contacts to the address book. But these situations may be outweighted by situations where the concept changes after ever menu interaction. For instance, a user may use the phone to make a call then leave the phone for a few hours before looking up someones work number. In this case, only one example is provided for the concept representing "making a call" before it is changed to a concept representing "looking up work number". Any learner will require a concept to remain stable for more than one training example, and therefore all the hypothesis-based approaches will require that a concept remain stable for more than one menu interaction.

To better evaluate the menu prediction approaches, we gathered data from 5 different users performing the same task in the *Contacts* application. Asking each user to perform a task creates a situation in which concepts remain stable for more than one menu interaction. Also, evaluating the menu selection prediction approaches according to the same task allows a fairer comparison to be made between the approaches. The task provided to each user consisted of: adding 11 new contacts, sending an SMS to 4 contacts in a particular group and deleting 2 contacts. The results are shown in Figure 5.

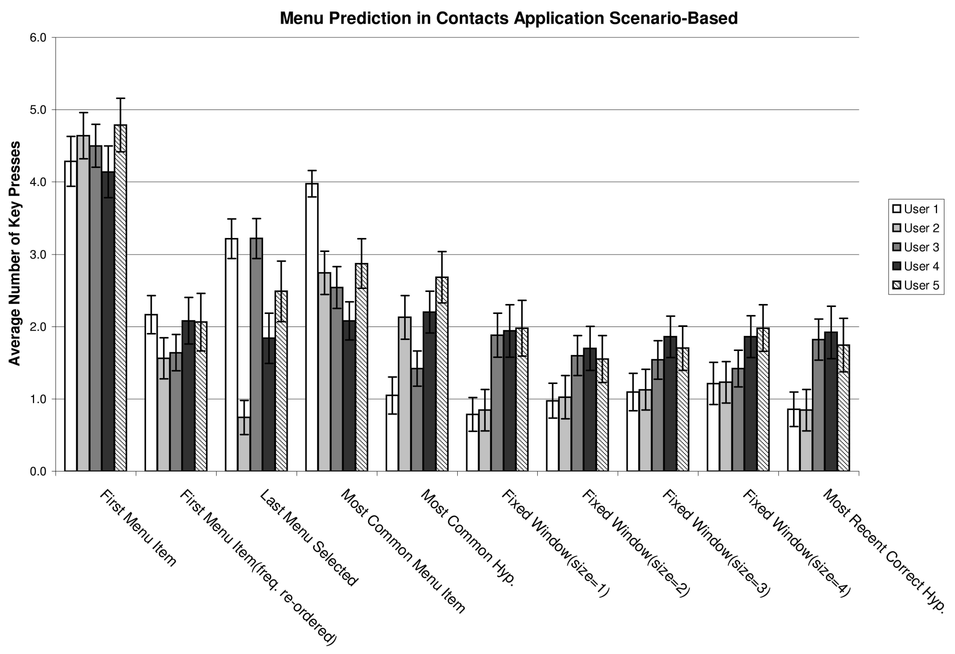

Fig. 5. Menu prediction strategies evaluated on 5 users performing the same task in the Nokia^{TM}Contacts application.

It is apparent from Figure 5 that the *First Menu Item* approach can be improved upon, in some cases to the extent of saving the user 3 key presses per menu selection [3]. The variability exhibited by the *First Menu Item(frequency re-ordered)*, *Last Menu Selected*, *Most Common Menu Item* and *Most Common Hypothesis* approaches suggests that they are not general solutions to menu selection prediction for all users. We also see that no one approach performs best for all users. This could suggest the need for a hybrid approach, in which the approach used is determined by the user.

All four *Fixed Window* approaches and the *Most Recent Correct Hypothesis* approach performed well on the data. Furthermore, the average number of key presses was consistently low for each user with these approaches. This suggests that not only can identifying a concept be useful, but that some method of handling concept changes is required.

[3] Over the entire task, our *Most Recent Correct Hypothesis* approach when compared with the *First Menu Item* approach, provided a saving of 144, 148, 164, 111 and 143 key presses for users 1, 2, 3, 4 and 5 respectively.

The performance of the four *Fixed Window* approaches and our *Most Recent Correct Hypothesis* approach was similar for each user. Although the performance of these approaches was similar, the *Fixed Window* approaches differ from our *Most Recent Correct Hypothesis* approach by considering all previous examples(of a fixed window size) relevant when evaluating a hypothesis. A deficiency of using a fixed window approach is when a concept change occurs. In these cases it is possible for examples generated from a previous concept to remain in the window and be considered relevant. This can be seen in Figure 3. The effects of a fixed window size was not obvious from Figure 5. This may have been due to the fact that the concepts we examined all remained stable for a few menu interactions, and therefore were able to be tracked by small window sizes. However, if the concepts all varied in the period of time they remained stable, then a window of fixed size would invariably contain not enough or too many training examples. Our *Most Recent Correct Hypothesis* approach in effect assigns to each hypothesis an adaptive sized window of training examples. Whenever a hypothesis becomes inconsistent with a new example, its window is cleared of examples. By not considering examples generated from a previous concept when evaluating the hypothesis space, we provide a more rational approach to hypothesis selection.

7 Conclusion

The idea of menu selection prediction has allowed machine learning to be incorporated into a mobile phone user interface. This increases the efficiency of the interface while mainting its predictability. The menu selection prediction approach discussed in Section 3 was shown through simulation to be highly suitable for the intended learning task, which was characterised by frequent concept changes. Also, an implementation of the menu selection prediction approach on a mobile phone demonstrated its feasibility.

References

1. M. Byrne. ACT-R/PM and menu selection: Applying a cognitive architecture to HCI. *International Journal of Human-Computer Studies*, 55:41–84, 2001.
2. A. Cypher. Eager: Programming repetitive tasks by example. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 33–39, 1991.
3. B. Davison and H. Hirsh. Predicting sequences of user actions. In *Predicting the Future: AI Approaches to Time-Series Problems*, pages 5–12, Madison, WI, July 1998. AAAI Press.
4. B. Davison and H. Hirsh. Probabilistic online action prediction. In *Intelligent Environments: Papers from the AAAI 1998 Spring Symposium, Technical Report SS-98-02.*, pages 148–154, 1998.
5. P. Gorniak and D. Poole. Predicting future user actions by observing unmodified applications. In *AAAI/IAAI*, pages 217–222, 2000.
6. S. Greenberg and I. Witten. Adaptive personalized interfaces- a question of viability. *Behaviour and Information Technology*, 4(1):31–45, 1985.

7. M. Harries, C. Sammut, and K. Horn. Extracting hidden context. *Machine Learning*, 32(2):101–126, 1998.
8. L. Hermens and J. Schlimmer. A machine learning apprentice for the completion of repetitive forms. *IEEE Expert*, 9:28–33, 1994.
9. R. Klinkenberg. Learning drifting concepts: example selection vs. example weighting. *Intelligent Data Analysis, Special Issue on Incremental Learning Systems Capable*, 8(3):281–300, 2004.
10. B. Korvemaker and R. Greiner. Predicting unix command lines: Adjusting to user patterns. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 230–235. AAAI Press / The MIT Press, 2000.
11. M. Kubat. Floating approximation in time-varying knowledge bases. *Pattern Recognition Letters*, 10:223–227, 1989.
12. P. Langley. Machine learning for adaptive user interfaces. In *KI - Kunstliche Intelligenz*, pages 53–62, 1997.
13. P. Langley. User modeling in adaptive interfaces. In *Proceedings of the Seventh International Conference on User Modeling*, pages 357–370. Banff, Alberta: Springer, 1999.
14. C. Lindholm and T. Keinonen. *Mobile Usability: How Nokia Changed the Face of the Mobile Phone*. McGraw-Hill Professional, New York, NY, USA, 2003.
15. J. Mitchell and B. Shneiderman. Dynamic versus static menus: an exploratory comparison. *SIGCHI Bulletin*, 20(4):33–37, 1989.
16. T. Mitchell, R. Caruana, D. Freitag, J. McDermott, and D. Zabowski. Experience with a learning personal assistant. *Communications of the ACM*, 37(7):80–91, 1994.
17. E. Nilsen. *Perceptual-motor control in human-computer interaction*. PhD thesis, University of Michigan, 1991.
18. M. Salganicoff. Tolerating concept and sampling shift in lazy learning using prediction error context switching. *Artificial Intelligence Review*, 11(1-5):133–155, 1997.
19. J. Schlimmer and R. Granger. Incremental learning from noisy data. *Machine Learning*, 1(3):317–354, 1986.
20. J. Schlimmer and P. Wells. Quantitative results comparing three intelligent interfaces for information capture: A case study adding name information into an electronic personal organizer. *Journal Artificial Intelligence*, 5:329–349, 1996.
21. A. Sears and B. Shneiderman. Split menus: effectively using selection frequency to organize menus. *ACM Transactions on Computer-Human Interaction*, 1(1):27–51, 1994.
22. R. St. Amant, T. Horton, and F. Ritter. Model-based evaluation of cell phone menu interaction. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 343–350, 2004.
23. G. Widmer. Combining robustness and flexibility in learning drifting concepts. In *Proceedings of the European Conference on Artificial Intelligence*, pages 468–472, 1994.
24. G. Widmer and M. Kubat. Learning flexible concepts from streams of examples: FLORA 2. In *Proceedings of the European Conference on Artificial Intelligence*, pages 463–467, 1992.
25. G. Widmer and M. Kubat. Effective learning in dynamic environments by explicit context tracking. In *Proceedings of the European Conference on Machine Learning*, volume 667, pages 227–243. Springer-Verlag, 1993.