Hidden Markov model finds behavioral patterns of users working with a headmouse driven writing tool

Gy. Hévízi, M. Biczó, B. Póczos, Z. Szabó, B. Takács and A. Lőrincz
Department of Information Systems
Eötvös Loránd University
Budapest, H-1117
E-mail: lorincz@inf.elte.hu

***Abstract*— We studied user behaviors when the cursor is directed by a head in a simple control task. We used an intelligent writing tool called Dasher. Hidden Markov models (HMMs) were applied to separate behavioral patterns. We found that a similar interpretations can be given to the hidden states upon learning. It is argued that the recognition of such general application specific behavioral patterns should be of help for adaptive human-computer interfaces.**

I. Introduction

The navigational, data and text entry tools and their combinations are gaining importance in today's popular field of human-computer interfaces (HCI). As computers show off more processing power, these tools are becoming more sophisticated: finally, adaptive features are finding their way into conventional interfaces. For a survey on current intelligent user interfaces see [1].

Any form of intelligence built in these interfaces inevitably assumes that the computer should be aware of the user's actual state. This implies that some form of a behavioral model should be implemented. In the last two decades well-known classical mathematical modeling methods were applied for this purpose. The practical targets of these efforts are extremely broad.

A. Previous Works

A straightforward solution for describing the behavior in time is the Markov chain (MC) model. MC is a simple state space model where stochastical transitions happen between states supposing the Markov property, which means that the probability of getting into a specific next state depends on the current state only. First and second order Markov chains were built from user customs to predict user requests for Web pages [2]. On the field of data mining and knowledge discovery, probabilistic regular grammars were employed to extract user navigation patterns [3]. Note that these grammars are related to Markov chains. The page visiting history of the user is analyzed by a special grammar where the strings generated with a higher probability correspond to the user's preferred trails.

Cadez at al. examines human behavior in a 'digital environment', e.g., the Web [4]. They developed a model of mixtures of Markov chains to cluster and visualize human navigation. Their method based on a high level classification of the visited pages (e.g. news, tech, weather, etc.). They applied it for determining the browsing patterns of the visitors of a commercial web site. A parsimonious and generative representation of behavior was created to describe a group of individuals by Girolami et al [5]. Their approach assumes that the manifestation of human expressions is a production of different underlying generating processes, thus a mixture of Markov chains should be able to characterize the heterogeneous features of human behavior in a suitable way. The distributed and dynamic model based on latent Dirichlet allocation is applied for analyzing text editor and telephone usage and to map Web browsing behavior.

A direct generalization of Markov chains are the hidden Markov models (HMMs) [6], [7]. This approach presumes that we are unable to observe the exact states, only a stochastic function of these (hidden) states is available to us. HMMs were used by Lane [8] for detecting computer usage anomalies when searching for stolen computer accounts. A HMM is constructed for the normal working state of the computer users, and by using the HMMs for classification, it is possible to detect deviations from the expected behavioral pattern. The same classification ability was exploited for recognizing human gestures [9], [10].

Bayesian networks[1] are related to these models. For an explanation on Bayesian networks the reader is referred to [11]. Dynamic Bayesian Networks (DBNs) are a more general formulation of HMMs. To predict real-time interaction behavior, DBNs with different topological forms were compared to the classical rule-based approach by [12].

A more practical work for assistance with modeling of users is the well-known Office Assistant, developed in the Lumiére project of the Microsoft Research Institute [13]. The application utilizes a Bayesian network to determine when the user needs help. An intelligent assistance system is described by Xiangyang et al. where uncertain and incomplete multiple modality sensory observation is used for assessing the user's current affective state [14]. A Bayesian approach is used by Conati for detecting emotional states in educational systems: socially intelligent, pedagogical agents are formed to entertain the user and to provide optimal learning content [15].

[1]Subtypes of Bayesian networks include casual networks, influence diagrams, belief networks, and relevance diagrams.

B. Our Motivations and Our Contribution

Our motivation is to promote direct adaptation to the user by means of reinforcement learning, where gain in performance is the 'goal' of the learning algorithm of the computer. Reinforcement learning concerns Markov decision processes. In turn, HMMs suit well our long term goal. Therefore, we would like to know if prototype behaviors emerge in particular tasks and if those behaviors can be discovered by HMMs. If the answer to this question is positive, then the route towards application specific user adaptation may become possible.

Recently, an interesting mouse-driven text entry solution called Dasher has been introduced [16]. We addressed the issue what kinds of behavioral patterns emerge upon practicing Dasher. Are these patterns similar for a set of users? Another question was how to describe these features in a more formal way and – possibly – how to model the actual behavior of the user.

Regarding the structure of the paper, first we briefly introduce the tools used in our experiments in Section II. In Section III a technical description of the conducted experiments is provided and the results of our efforts are presented. These results are discussed in Section IV. Conclusions are drawn in Section V.

II. Tools

A. Dasher

Dasher has been developed by the Inference Group of the Cavendish Laboratory of Cambridge University[2]. Dasher is driven by pointing gestures. It has a zooming interface: you point to your region of interest, and the display zooms to that region. The display is painted with a single letter or with a combination of letters, so that any point corresponds to a piece of text. Typing is achieved by choosing the appropriate letter or combinations of letters. Dasher is equipped by a predictive language model. It forecasts the probabilities of the possible next letters or letter combinations. Probable pieces of text are made larger and can be selected easily. Improbable pieces of text (for example, text with spelling mistakes) are made smaller, so they are harder to write.

We used a slightly modified version of the 3.0.1 version of the application where the language model can be turned off. Two screenshots of the application are shown in Figure 1.

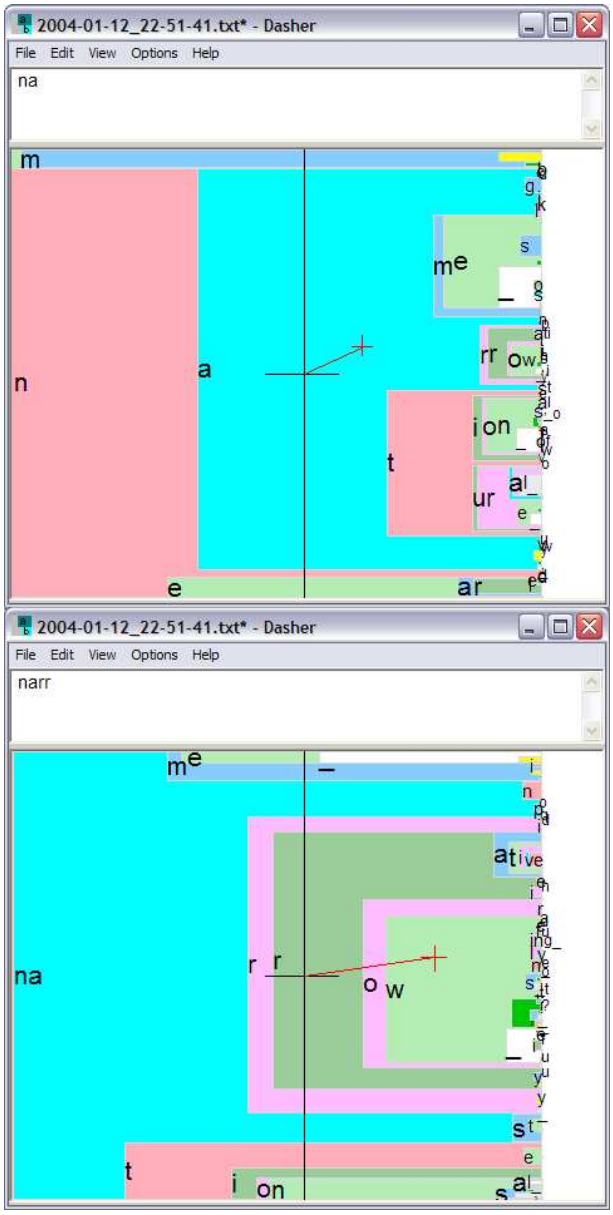

Fig. 1. **Dasher in action.** The upper screenshot was taken when the user started to type the word 'narrow'. The lower snapshot was taken about a second later. The reader should imagine a continuous transformation which happens between the two screenshots. On the upper one, the user had already typed the letters 'n' and 'a', and he/she had been advancing towards the end of the word. On the lower picture, two 'r's were 'typed'. Letters and combinations of letters emerge on the right side of the screen, while the whole board is sliding to the left. The typed letters assemble on the left hand side. The direction and the speed of the sliding is determined by the position of the cursor (the red cross on the figure). Selecting letters of the alphabet is possible by moving the cursor vertically. Horizontal position of the cursor influences the actual speed of the text flow. Notice that probable combinations have larger areas but they are pushed aside as the movement of the user makes clear that he/she intends to type something else.

B. Input Devices

We developed a special mouse system which is controlled by head movements and can be downloaded for free[3]. The headmouse combines head detection built on Haar-wavelets and a tracking solution based on optic flow. The result is a non-intrusive and cheap (only a webcam is needed) interaction tool. Head and eye driven tools should gain importance in the future because the fast development of mobile devices requires alternative solutions in different environments. Headmouse systems are already important for augmented and alternative

[2]http://www.inference.phy.cam.ac.uk/is/

[3]http://www.nipg.inf.elte.hu/headmouse/headmouse.html

communication for disabled people. For control experiments, the normal desk mouse was used.

C. Hidden Markov Model (HMM)

We intended to use the components of the speed vector of the cursor as the observed variables. That implies that we are expecting to find behavioral components which are patterns of the actual *user movement*. The user's intended course of the movement is distorted because there is noise in the input process and unwanted movement of the head may also occur. We can assume that the actual user movement is a noisy observation of underlying behavioral components. Therefore choosing a hidden Markov model for the underlying mathematical model is justified.

Training of the HMM [6] was achieved by inputting the observed variables and by tuning the parameters of the HMM until it could generate the observed variables with the highest probability. In this procedure Maximum Likelihood (ML) estimation was applied. Technically, we have two observed variables here: one for the movements in the vertical direction and another one for the horizontal direction. Provided that the emission had Gaussian noise in the different hidden states, the analysis should result in hidden states whose emission functions correspond to different cursor speed regions. The center of the region corresponds to the most probable cursor speed in that state. This probability decreases monotonically as a function of the distance from the center of the region. We will characterize any hidden state with its center and its variance ellipse representing the corresponding emission probability density.

We would like to emphasize that the utilized temporal resolution – defined by the cursor sampling rate – does not enable recognizing events (states) on a time scale longer than some hundreds of a millisecond. By aggregating cursor movements in longer temporal regions it is possible to extract states with longer duration. Information on user intentions, however, can not be provided without more sophisticated methods, which, for example, consider possible text-fragments that have been typed and may be typed given the language model.

III. Experiments and Results

Experiments were conducted with five volunteer Ph.D. students. They did not have previous experiences neither with Dasher nor with the headmouse. Their task was to type randomly selected short English sentences, created from lyrics of different songs (for example 'children need traveling shoes'). We saved the cursor trajectories with a 30-50 Hz sampling rate (the rate depended on the computational load of the computer). First, 25 training experiments were conducted in five days. After this, the participants had the opportunity to improve their skills for three weeks. Then the survey was repeated with a fewer number of experiments. By analyzing the results in terms of typing speed we can conclude that four out of the five users gained good typing skills in about six to seven experiments. The final performance reached the typing speed of an average computer user for all participants (Fig.2).

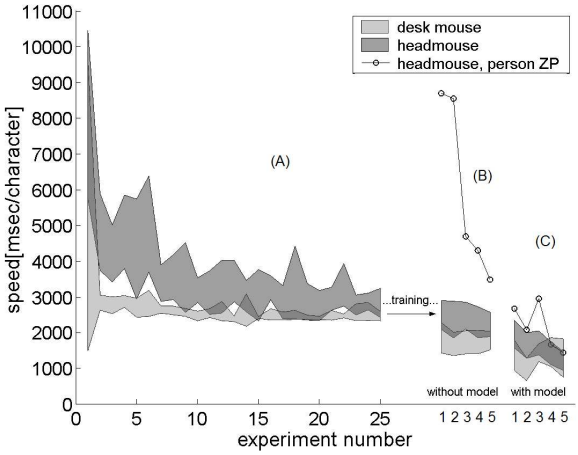

Fig. 2. **Experiments with Dasher using conventional desk mouse (light gray) and headmouse (dark gray)**. Average time required for inputting one character vs. experiment number. Width of stripes: standard deviation regions for four participants. Overlap between STD regions for different methods have an intermediate color. Performance of the fifth participant (ZP) is classified as an outlier and is plotted separately (dashed line in ranges B and C). Up to the 25th experiment (range A): training without intelligent English language predictor. Duration of range A: 5 days. Further training took place for 3 weeks. Section B: 5 experiments after training. Section C: 5 experiments with the intelligent English language predictor for the first time. Data were taken on the same day when range B was recorded. Results are presented in temporal order in every range.

One of the participants showed lower performance at the beginning, but in the later phase of the training he/she reached the performance of the others. This subject is referred to as 'ZP'.

A HMM with Gaussian emission functions was trained for every participant. The training made use of the Expectation-Maximization (EM) algorithm [17].

Human head movements tend to build from strictly horizontal or vertical elements, therefore we considered the observations as being independent in the x and y directions. With other words, we prescribed a diagonal form for the covariance matrices of the observations, i.e. the axes of the variance ellipses are parallel to the x and y axes of the screen. We also tried a circular form or an arbitrary ellipse, but the most informative results was found by the diagonal setup. We also found that the diagonal form is the most appropriate for recognition tasks, i.e. models built with this kind of restriction have got the best recognition capabilities.

To select an appropriate number of states we scanned a realistic domain of this parameter in the analysis. HMMs were trained with different state numbers; 3, 4, 5, 8 and 15. We found that 3 or 4 states are not expressive enough for characterizing different behaviors and important details were missed. When choosing a high number of states, the HMM tends to have an intriguing log-polar structure, i.e. there are many states around the origin with small variances and a smaller number of states are formed further out from the

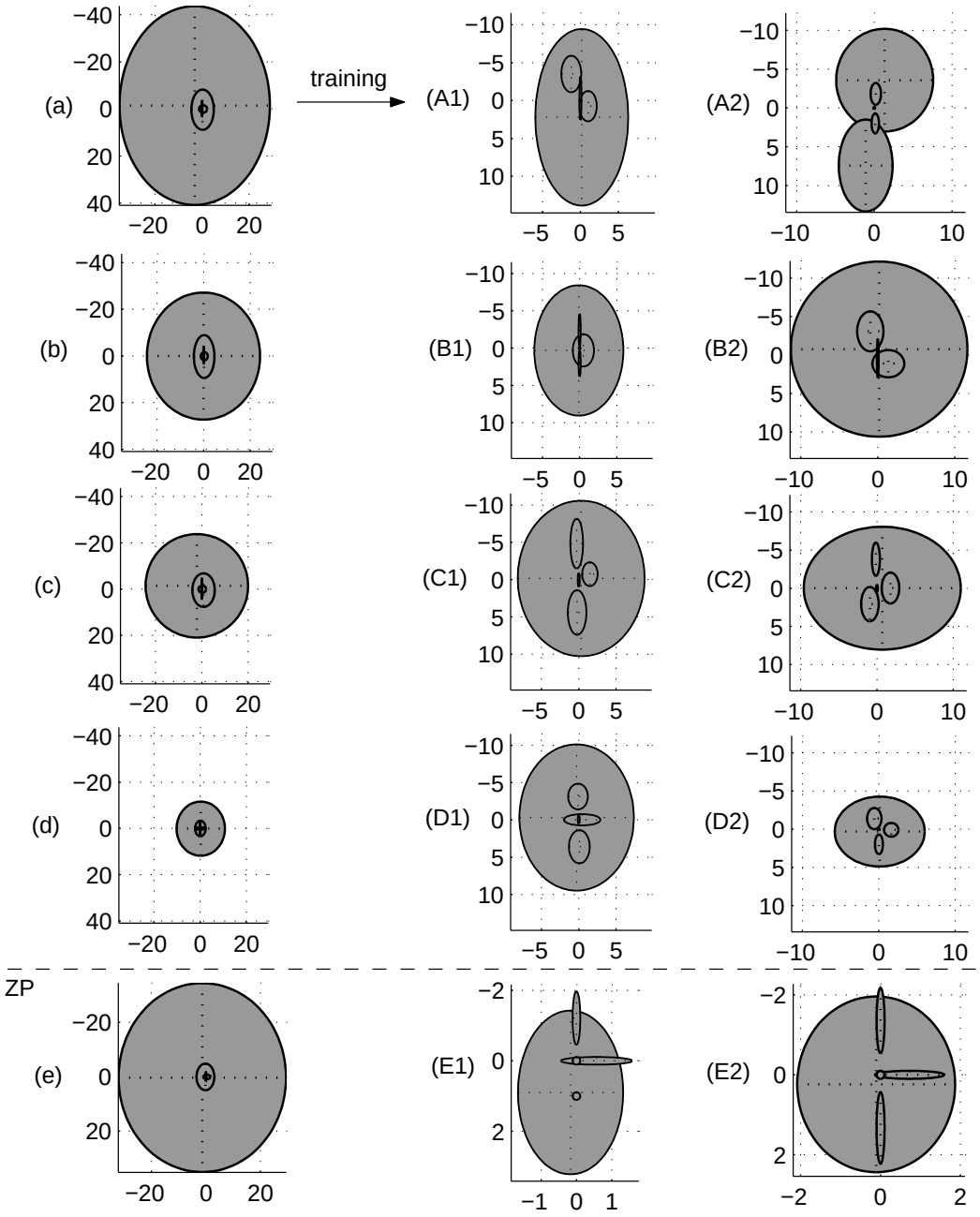

Fig. 3. **Hidden states found by the HMM analysis.** Results for 5 hidden states are presented. Left column (a-e): variance ellipses of the Gaussian emission functions of the hidden states, from the 'inexperienced' time region (first 15 experiments of range A of Fig. 2). Middle column: hidden states extracted from range B of Fig. 2. Right column: hidden states extracted with language model turned 'on' (range C of Fig. 2). Note that the ellipses of the first row and the last row have different characteristics. Note also the different scale of the last row.

origin. These latter states have higher variances. Although more details about the movement of the user can be determined by such HMM-based 'discretization', but our interpreting capability of the HMM states was lost.

Uniform probability for the initial transition matrices was set and the starting states were initialized to equal probabilities. The parameters of the observation function (i.e., the expected value and the variance) was initialized randomly by the k-means algorithm. The EM algorithms can be trapped in local minima, therefore we started the EM algorithm 10 times using different starting conditions to overcome this problem. The most probable models as defined by ML were selected for every user. The most probable model, by definition, reproduces the user's inputs with the highest probability.

Data sets were split. The first phase of the experiment series contains the first 25 experiments (Fig. 2, range denoted by A). Users are considered 'inexperienced' up to experiment number 15, later they are called 'experienced' users. We trained models

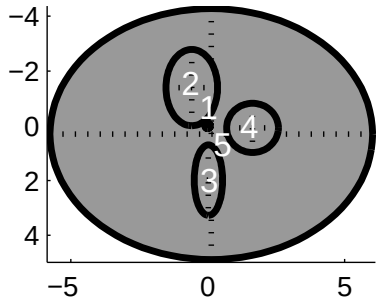

Fig. 4. **Intuitive labeling of the hidden states.** (1) 'It is going fine, keep it steady', (2) 'I need to choose a letter (or word fragment), which is closer to the beginning of the alphabet', (3) 'I need to choose a letter (or word fragment), which is closer to the end of the alphabet', (4) 'I can accelerate the typing speed', (5) 'I made a mistake, correction is needed fast'.

for every user for 1) the 'inexperienced' part of section A, 2) for section B and 3) for section C of the data. The results can be seen in Fig. 3. It is obvious that the more experienced the user, the more separated and more distinctly arranged the hidden states become. Moreover, the hidden states of each participant have certain similarities that we shall discuss later.

IV. Discussion

A. Interpretation of the hidden states

In case of Dasher, models with 5 hidden states have an important property: they catch something general for *all* users. This encourages us to map these states to behavioral components. We can always find

(1) a state which is concentrated on the origin and has a small variance (this is barely visible on some of the figures),
(2) a state with a moderate variance situated above the origin,
(3) a state with a moderate variance situated below the origin,
(4) in four cases, a state with a moderate variance situated to the right the origin,
(5) and a state (or two states in one case), which has (have) a large variance.

Let us recall the effect of mouse movements in Dasher. Vertical movements are related to changing the selection of letters of the alphabet: Going up (down) aims to find a letter closer to letter 'a' ('z') in the alphabet. Horizontal movements change the typing speed. A natural interpretation for the states formed can be provided for Fig.3(C2)-(E2) as it is shown in Fig. 4.

States of Fig. 4(A2) have got a slightly different interpretation, because this user kept an optimal typing speed, no accelerating state was distinguished and the mistake state was resolved into two distinct states, which may not be mistakes at all:

1) 'It is going fine, keep it steady',
2) 'I need to choose a letter (or word fragment), which is somewhat closer to the beginning of the alphabet',
3) 'I need to choose a letter (or word fragment), which is somewhat closer to the end of the alphabet',
4) 'I need to choose a letter (or word fragment), which is much closer to the beginning of the alphabet',
5) 'I need to choose a letter (or word fragment), which is much closer to the end of the alphabet'.

Similar pattern started to emerge in the second row of Fig. 3.

The interpretations are also reinforced by Fig. 3: after training, the size of the ellipse with the largest variance, has become significantly smaller. This phenomenon marks an increase in performance, the earlier detection of necessary corrections and the smaller motion speeds required for corrections. For Fig. 3(A2), it seems that the correcting state has disappeared. We shall return to this point later. For the case of participant 'ZP' (Fig. 3(e), (E1) and (E2)), it was found that 'ZP' tried to avoid making any error and developed rather distinct behavioral patterns. Participant 'ZP' is the best example for the interpretation on Fig. 4.

B. Transitions amongst behaviors

The detailed transitions of the five states are plotted on Figure 5 in the form of Hinton diagrams.

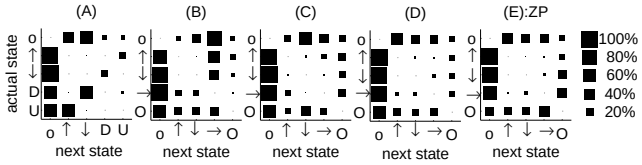

Fig. 5. **Hinton diagrams of transition probabilities for trained users.** Transition probabilities concern subfigures A2-E2 of Figure 3. States 1 to 5 of Fig. 4 are denoted by symbols (o) ↑, ↓, →, and (O), respectively. For the special arrangement of the first subject's A2 model, states are denoted by symbols (o) ↑, ↓, (D=down), and (U=up), where U and D marks the two states with bigger variance in the corresponding position. Sizes of the squares are proportional to the corresponding transition probabilities. Transitions where the state remains unchanged are neglected (the diagonals contain zeros).

Time shares of specific states in percentage of all states were computed (Tables I-II). The Viterbi algorithm was used for this purpose and the most probable current states were determined. Notations is as follows: states 1 to 5 of Fig. 4 are denoted by symbols (o) ↑, ↓, →, and (O), respectively. For the special arrangement of Fig. 3(A2): states are denoted by symbols (o) ↑, ↓, (D=down), and (U=up), where U and D marks the two states with the big variance in the corresponding position.

TABLE I
PERCENTAGE OF HIDDEN STATES – NO LANGUAGE MODEL
FOR NOTATIONS, SEE FIGURE 3 AND 5

code	o	↑	↓	→ or D	O or U
A1	38.4	7.1	30.6	14.5	9.4
B1	47.1	17.7	16.4	12.1	6.7
C1	44.7	15.5	17.6	9.1	13.1
D1	35.2	18.3	17.1	19.4	10.1
E1	76.4	9.0	5.4	3.8	5.4

Subject 'ZP' spends considerably less time in *Mistake State* than any other participant. Considering that 'ZP' has reached the performance of the others at the end of study, and that 'ZP' has extremely well articulated hidden states, one may

TABLE II

PERCENTAGE OF HIDDEN STATES – WITH LANGUAGE MODEL

FOR NOTATIONS, SEE FIGURE 3 AND 5

code	o	↑	↓	→ or D	O or U
A2	41.2	25.8	17.0	5.7	10.3
B2	48.0	7.1	28.7	11.1	5.1
C2	35.1	12.6	21.2	14.2	16.9
D2	43.0	20.3	11.9	12.1	12.7
E2	60.8	12.2	10.3	9.0	7.7

conclude that 'ZP' has applied an error avoiding safe typing strategy during learning.

On the other hand, the Hinton diagram reveals that the first participant applied a strategy forming two typical series during a large proportion of the time. Series 1: (D) $\Rightarrow \downarrow \Rightarrow$ (o). Series 2: (U) $\Rightarrow \uparrow \Rightarrow$ (o). From *Good State* transitions occurred to any other states, whereas state (U) is basically never followed by either state (D) or state $\downarrow$ and similarly, state (D) is basically never followed by state (U) or state $\uparrow$. This observation provides further support to our impression that the first participant succeeded in avoiding the *Mistake State*. Small adjustments were utilized or large correcting movements were followed by small adjustments in this case.

V. CONCLUSIONS AND OUTLOOK

In this work, experiments with writing tool Dasher were conducted. We studied the behavior of the users before and after the training using a hidden Markov model. The hidden states had similar arrangements for every user, and we can label these states with different behavioral components in an intuitive way.

The recognition of the users' behavioral patterns may promote the development of adaptive human-computer interfaces in the future. As described in the ISO 18529 standard, a human-centered design process for an interactive system starts with the identification of the target user group(s) by determining group characteristics, and then continues by developing a different instance of the software design for every possible context of user groups. Our view is that this design process can be, and in some cases, should be postponed. It can be postponed when on-line adaptive functionality, like behavioral component recognition is available. It should be postponed, when all types of behavioral patterns, such as strategies followed by very cautious users or by dyslexic persons can not be foreseen. Such adaptive tools can be used for different purposes, e.g. (i) as a tool for user clustering, or possibly for user identification [18], (ii) as a remote diagnostic tool, and/or (iii) as the underlying technology for performance optimization or personalization for individual users. Considering case (iii), the identification of behavioral components may offer alternative help options, e.g., limiting cursor speed adaptively to avoid mistakes, etc. More complex interaction can be envisioned by employing reinforcement learning to optimize the actions of the computer.

The HMM technology is promising, but predictions on longer time scales seem necessary for better service to the user. Such longer predictions can be achieved by including information about the actual task, such as the language model in the case of typing tool Dasher. This is an obvious extension of the model, which however, may lead to combinatorial explosion. This point deserves further investigations.

REFERENCES

[1] P. Ehlert, "Intelligent user interfaces: introduction and survey," 2003. [Online]. Available: citeseer.nj.nec.com/ehlert03intelligent.html

[2] I. Zukerman, D. Albrecht, and A. Nicholson, "Predicting users' requests on the www," 1999. [Online]. Available: citeseer.nj.nec.com/zukerman99predicting.html

[3] J. Borges and M. Levene, "Data mining of user navigation patterns," in *WEBKDD*, 1999, pp. 92–111. [Online]. Available: citeseer.nj.nec.com/borges00data.html

[4] I. V. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White, "Visualization of navigation patterns on a web site using model-based clustering," in *Knowledge Discovery and Data Mining*, 2000, pp. 280–284. [Online]. Available: citeseer.nj.nec.com/cadez00visualization.html

[5] M. Girolami and A. Kabán, "Sequential activity profiling: Latent dirichlet allocation of markov chains," December 2003, seventeenth Annual Conference on Neural Information Processing (NIPS2003), Vancouver, British Columbia, Canada.

[6] L. R. Rabiner and B. H. Juang, "An introduction to hidden Markov models," *IEEE ASSP Magazine*, pp. 4–15, January 1986.

[7] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[8] T. Lane, "Hidden markov models for human/computer interface modeling," 1999. [Online]. Available: citeseer.nj.nec.com/lane99hidden.html

[9] G. Rigoll, A. Kosmala, and S. Eickeler, "High performance real-time gesture recognition using hidden Markov models," *Lecture Notes in Computer Science*, vol. 1371, 1998. [Online]. Available: citeseer.nj.nec.com/rigoll98high.html

[10] S. Eickeler, A. Kosmala, and G. Rigoll, "Hidden Markov model based continuous online gesture recognition," in *Int. Conference on Pattern Recognition (ICPR)*, Brisbane, 1998, pp. 1206–1208. [Online]. Available: citeseer.nj.nec.com/eickeler98hidden.html

[11] M. J. Jordan, Ed., *Graphical models*. Cambridge MA: MIT Press, Cambridge, 1999.

[12] A. Kuenzer, C. Schlick, F. Ohmann, L. Schmidt, and H. Luczak, "An empirical study of dynamic bayesian networks for user modeling," 2001. [Online]. Available: citeseer.nj.nec.com/kuenzer01empirical.html

[13] E. Horvitz, J. Breese, D. Heckerman, D. Hovel, and K. Rommelse, "The lumiere project: Bayesian user modeling for inferring the goals and needs of software users," 1998. [Online]. Available: citeseer.nj.nec.com/horvitz98lumiegravere.html

[14] X. Li and Q. Ji, "Active affective state detection and assistance with dynamic bayesian networks," 3rd Workshop on Affective and Attitude User Modeling Assessing and Adapting to User Attitudes and Affect: Why, When and How? in conjunction with User Modeling, June 2003, pittsburgh, PA, USA.

[15] C. Conati, "Probabilistic assessment of user's emotions during the interaction with educational games," 2002. [Online]. Available: citeseer.nj.nec.com/502106.html

[16] D. J. Ward, "Adaptive computer interfaces," Ph.D. dissertation, Churchill College, Cambridge, 2001.

[17] A. P. Dempster, N. P. Lair, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statist. Soc.*, vol. B 39, no. 1, pp. 1–38, 1977.

[18] J. Stone, "Face recognition: When a nod is better than a wink," *Current Biology*, vol. 11, pp. R663–664, 2001.