

Modeling Temporal Behavior via Structured Hidden Markov Models: an Application to Keystroking Dynamics

Ugo Galassi, Attilio Giordana, Charbel Julien, and Lorenza Saitta

Dipartimento di Informatica, Università Amedeo Avogadro
Via Bellini 25G, Alessandria, Italy
{ugo.galassi, attilio.giordana, charbel.julien, lorenza.saitta}@unipmn.it

Abstract. Structured Hidden Markov Models (S-HMM) are a variant of Hierarchical Hidden Markov Models; it provides an abstraction mechanism allowing a high level symbolic description of the knowledge embedded in S-HMM to be easily obtained, at the same time reducing the complexity of using and learning the model.

S-HMMs are particularly well suited to build up profiles of discrete processes, described by meaningful sequences of symbols possibly interleaved with *gaps*, i.e., subsequences whose useful information resides in their duration and not in their content.

In this paper we will first introduce the model, and then we will concentrate on the description of an application, namely the characterization of biometric sequences (keyboard stroke duration) used for an identification task in computer access.

Key words: Hidden Markov Model, keystroking dynamics, user authentication.

1 Introduction

Modeling an agent behavior is a task that is becoming of primary relevance in computer networks. In fact, critical applications, such as intrusion detection [11, 18, 6], network monitoring [19, 12], or environment surveillance [4, 17], are heading towards approaches in which modeling processes and users is a key to success.

However, to automatically capture a model of a behavior is not a trivial task. On the one hand, the features that the model should capture, and the ones that should be ignored, are often problem specific. On the other hand, current formal modeling tools become rapidly intractable when the complexity of the patterns to be captured increases. Then, only simple technologies can be used in practice.

Problems of modeling an agent behavior fall into two classes. The first class contains problems where a stationary, long-term behavior has to be modeled, for instance, the daily usage of a set of resources [11]. In this case, building a behavior model can be frequently reduced to collecting statistics related to

some set of global variables, and the particular sequences in which actions are performed is not important. The second class contains more intriguing problems, aimed at explicitly capturing transitory behavior involving events duration and ordering; for instance, how long it takes for an agent to go from A to B, or what are the actions made by the agent in A and B. Modeling such behavioral aspects requires modeling time and action sequences, a problem for which still no general solution exists.

We focus on this second class of problems, and we propose a variant of Hidden Markov Models (HMM) [16], which we call *Structured Hidden Markov Models* (S-HMM), as a general modeling tool. In fact, an S-HMM has sufficient expressiveness to cope with most practical problems, while its complexity can be kept tractable.

In the following we provide a brief introduction to S-HMMs, referring the reader to other papers for the formal analysis of their properties [9, 8]. Then we will investigate the problem of modeling time duration and transitories using specific HMMs. Finally we will describe an application to keystroking dynamics, where modeling durations was the most critical issue.

2 The Structured HMM

The basic assumption underlying an S-HMM (see Bouchaffra and Tan [1]) is that a sequence $O = \{o_1, o_2, o_3, \dots, o_T\}$ of observations could be segmented into a set of subsequences O_1, O_2, \dots, O_N , each one generated by a sub-process with only weak interactions with its neighbors. This assumption is realistic in many practical applications, such as, for instance, speech recognition [16, 15], and DNA analysis [5]. S-HMMs aim exactly at modeling such kind of processes, and, hence, they are represented as directed graphs, structured into sub-graphs (*blocks*), each one modeling a specific kind of sub-sequences.

Informally, a block consists of a set of states, only two of which (the *initial* and the *end* state) are allowed to be connected to other blocks. Then, an S-HMM can be seen as a forward acyclic graph with blocks as nodes. As an S-HMM is itself a block, a nesting mechanism is immediate to define.

2.1 Structure of a Block

In this section, a formal definition of S-HMM will be provided. Adopting the notation used in [16], O will denote a sequence of observations $\{o_1, o_2, \dots, o_T\}$, where every observation o_t is a symbol v_k chosen from an alphabet V . An HMM is a stochastic automaton characterized by a set of states Q , an alphabet V , and a triple $\lambda = \langle A, B, \pi \rangle$, being:

- $A : Q \times Q \rightarrow [0, 1]$ a probability distribution, a_{ij} , governing the transition from state q_i to state q_j ;
- $B : Q \times V \rightarrow [0, 1]$ a probability distribution, $b_i(v_k)$, governing the emission of symbols in each state $q_i \in Q$;

- $\pi : Q \rightarrow [0, 1]$ a distribution assigning to each state $q_i \in Q$ the probability of being the start state.

A state q_i will be said a *silent* state if $\forall v_k \in V : b_i(v_k) = 0$, i.e., q_i does not emit any observable symbol. When entering a silent state, the time counter must not be incremented.

Definition 1. A basic block of an S-HMM is a 4-tuple $\lambda = \langle A, B, I, E \rangle$, where $I, E \in Q$ are silent states such that: $\pi(I) = 1$, $\forall q_i \in Q : a_{iI} = 0$, and $\forall q_i \in Q : a_{Ei} = 0$.

In other words, I and E are the input and the output states, respectively. Therefore, a composite block can be defined by connecting, through a forward transition network, the input and output states of a set of blocks.

Definition 2. Given an ordered set of blocks $\Lambda = \{\lambda_i | 1 \leq i \leq N\}$, a composite block is a 4-tuple $\lambda = \langle A_I, A_E, I, E \rangle$, where:

- $A_I : \mathbf{E} \times \mathbf{I} \rightarrow [0, 1]$, $A_E : \mathbf{I} \times \mathbf{E} \rightarrow [0, 1]$ are probability distributions governing the transitions from the output states \mathbf{E} to the input states \mathbf{I} , and from the input states \mathbf{I} to the output states \mathbf{E} of the component blocks Λ , respectively.
- For all pairs $\langle E_i, I_j \rangle$ the transition probability $a_{E_i I_j} = 0$ if $j \leq i$.
- $I \equiv I_1$ and $E \equiv E_N$ are the input and output states of the composite block, respectively.

According to Definition 2 the components of a composite block can be either basic blocks or, in turn, composite blocks. In other words, composite blocks can be arbitrarily nested. Moreover, we will keep the notation S-HMM to designate non-basic blocks only.

As a special case, a block can degenerate to the *null block*, which consists of the start and end states only, connected by an edge with probability $a_{IE} = 1$. The *null block* is useful to provide a dummy input state I or a dummy output state E , when no one of the component blocks is suited to this purpose.

An example of S-HMM structured into three blocks $\lambda_1, \lambda_2, \lambda_3$, and two *null blocks* λ_0, λ_4 , providing the start and the end states, is described in Figure 1.

2.2 Estimating Probabilities in S-HMM

Well-known algorithms are used to solve typical problems arising in using HMMs; in particular, the *forward backward* algorithm computes the probability $P(O|\lambda)$ that a model λ generates the observation O . The *Viterbi* algorithm computes the most likely sequence of λ 's states that generates O . Finally, the *Baum-Welch* algorithm allows the parameters of λ (i.e., the matrices A and B and the vector π) to be estimated from sequences.

The mentioned algorithms exploit two sets of iteratively computed functions, α and β , plus other auxiliary functions, γ and ξ , defined on α and β . Functions α and β have an intuitive meaning. Function $\alpha_t(i)$ evaluates, for every time t

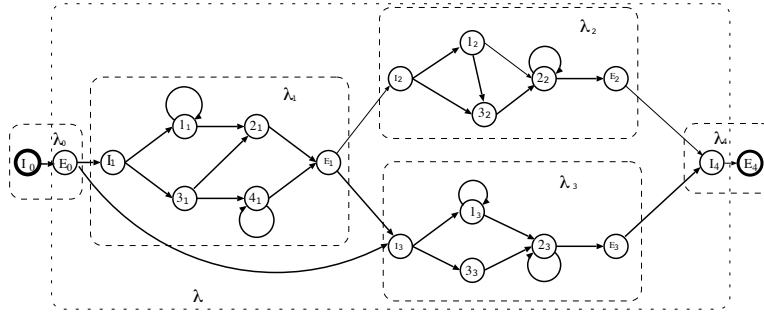


Fig. 1. Example of a Structured Hidden Markov Model, composed of three interconnected blocks, plus two *null blocks*, λ_0 and λ_4 , providing the start and end states. Distribution A is non-null only for explicitly represented arcs.

($1 \leq t \leq T$) and every state q_i ($1 \leq i \leq N$), the joint probability of being in state q_i and observing the symbols from o_1 to o_t . Funtion $\beta_t(i)$ is complementary to $\alpha_t(i)$, and computes, for every time t ($1 \leq t \leq T$) and every state q_i ($1 \leq i \leq N$), the conditional probability of observing the symbols $o_{t+1}, o_{t+2}, \dots, o_T$, given that q_i is the state at time t . Then, $\alpha_T(E) = \beta_1(I) = P(O|\lambda)$.

All these functions are defined on basic HMMs, but we have extended them to the case of nested S-HMMs [8], and provided algorithms that calculate them incrementally. In an S-HMM each block interacts with others only through the *initial* and *end* states. Then, the computation of the functions α and β inside one block does not affect the one in other blocks, and, hence, parameters inside each block can be estimated independently of the other ones, reducing the standard $O(TN^2)$ complexity to $O(T \sum_h N_h^2)$, where N_h is the number of states inside block λ_h and $N = \sum_h N_h^2$.

3 Modeling Motifs and Duration

The basic assumption is that the behavior of an agent periodically performs usually short sequences of actions, typical of the task it executes, interleaved with phases where the activity cannot be modeled, because it is non-repetitive. By analogy with the DNA sequences in molecular biology, we will call *motifs* such a kind of characteristic sequences of actions. In fact, under the previous assumptions, the problem of agent profiling presents a strong analogy to the problem of discovering and characterizing coding subsequences in a DNA chromosome.

A model for interpreting a sequence must be a *global* model, able to identify both interesting patterns that occur with significant regularity, and *gaps*, *i.e.*, regions where no regularities are found. Generating a global model of the sequence is important, because it allows inter-dependencies among motifs to be detected. Nevertheless, a global model must account for the distribution of the observations on the entire sequence, and hence it could become intractable. We

tamed this problem by introducing special basic blocks, designed to keep low the complexity of modeling the irrelevant parts of sequences.

In the following we address the following issues: (a) how to construct basic blocks modeling motifs; (b) how to construct models of gaps between motifs; (c) how to extract (insert) knowledge in readable form from (to) an S-HMM; (d) how to find the interpretation of a sequence.

3.1 Modeling Motifs

A motif is a subsequence frequently occurring in a reference sequence set. Motif occurrences may be different from one to another, provided that an assigned equivalence relation be satisfied. In the specific case, the equivalence relation is encoded through a basic block of an S-HMM. Many proposals exist for HMM architectures oriented to capture specific patterns. Here, we will consider the Profile HMM (PHMM), a model developed in Bio-Informatics [5], which well fits the needs of providing a readable interpretation of a sequence. The basic assumption underlying PHMM is that the different instances of a motif originate from a canonical form, but are subject to insertion, deletion and substitution errors.

As described in Figure 2, a PHMM has a left-to-right structure with a very restricted number of arcs. Moreover, it makes use of *typed* states: *Match* states, where the observation corresponds to the expectation, *Delete* states (silent states) modeling deletion errors, and *Insert* states, modeling insertion errors supposedly due to random noise. According to this assumption, the distribution of the observations in all insert states is the same, and it can be estimated just once.

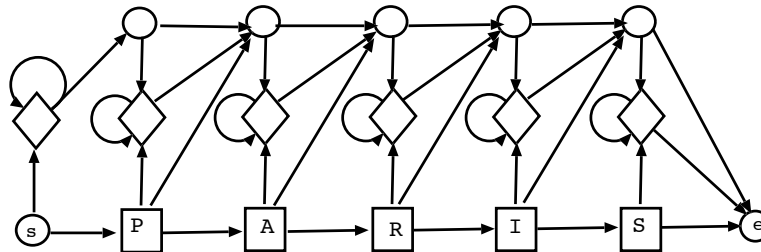


Fig. 2. Example of Profile Hidden Markov Model. Circles denote states with *no-observable* emission, rectangles denote *match states*, and diamond denote *insert states*.

After training, the canonical form can be easily extracted from a PHMM, by collecting the maximum likelihood observation sequence from the match states.

3.2 Modeling Duration and Gaps

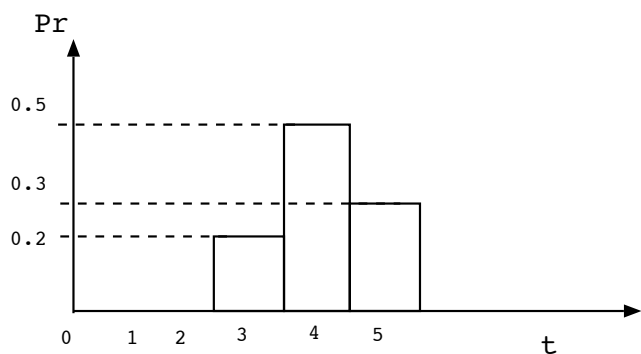
The problem of modeling durations arises when the time span covered by an observation or the interval length between two observations is important. In the

HMM framework, this problem has been principally faced in Speech Recognition and in Bio-informatics. However, the problem setting is slightly different in the two fields, and consequently the dominant approach tends to be different. In speech recognition, the input is a continuous signal, which, after several steps of signal processing, is segmented into variable length intervals, each one labeled with a symbol. Then, the obtained symbolic sequence is fed into a set of HMMs, which accomplish the recognition of long range structures, such as syllables or words, requiring thus to deal with interval durations [13]. In Bio-informatics the major application for HMMs is the analysis of DNA strands [5]. Here, the input sequence is a string of equal length symbols. The need of modeling duration comes from the presence of gaps, *i.e.*, substrings where no coding information is present. The gap duration is often a critical cues to interpret the entire sequence.

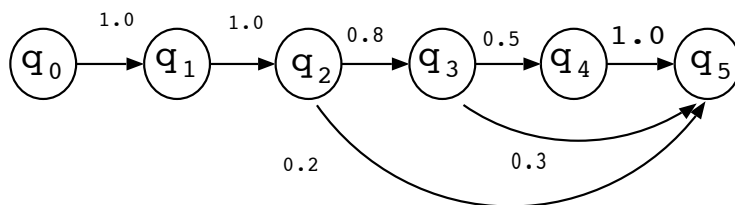
The approach first developed in Speech Recognition is to use Hidden Semi-Markov Models (HSMM), which are HMMs augmented with probability distributions over the state permanence [13, 10, 14, 21, 20]. An alternative approach is the so called *Expanded HMM* [10]. Every state, where it is required to model duration, is expanded into a network of states, properly interconnected. In this way, the duration of the permanence in the original state is modeled by a sequence of transitions through the new state network in which the observation remain constant. The advantage of this method is that the Markovian nature of the HMM is preserved. Nevertheless, the complexity increases according to the number of new states generated by expansion.

A similar solution is found in Bio-Informatics for modeling long gaps. In this case, the granularity of the sequence is given, and so there is no need of expansion. However, the resulting model of the gap duration is similar to the one mentioned above. A Profile HMM [5], naturally models the duration of observations according to the expansion technique, as shown in Section 4, but it is only able to model short gaps inside a motif, attributed to random noise controlled by a Poisson statistics. Nevertheless, single insertion states do not correctly model long gaps occurring in between two motifs. The most appropriate probability distribution for this kind of gaps may vary from case to case, but it is never the exponential decay defined by an insert state with a self-loop.

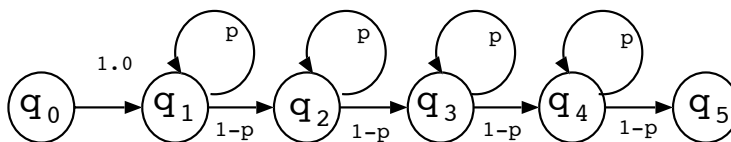
Several HMM topologies, proposed for modeling gaps, are reported in Figure 3. The architecture in Figure 3(b) can be used to model any duration distribution over a finite and discrete interval. However, the drawback of this model is the potentially large number of parameters to estimate. In all cases, the observation is supposed to be produced by random noise. Model in Figure 3(c) exhibits an Erlang's distribution, when the Forward-Backward algorithm is used. Unfortunately, the distribution of the most likely duration computed by Viterbi algorithm still follows an exponentially low. Therefore, this model, which is more compact with respect to the previous one, is not useful for the task of segmenting and tagging sequences by means of Viterbi algorithm. However, sequence segmentation and interpretation is not a primary task in the class of problems we are considering in this paper.



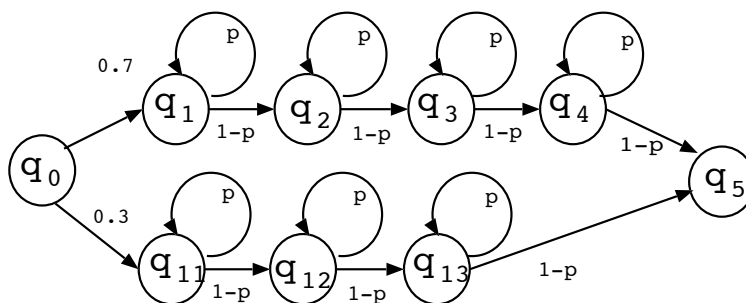
(a)



(b)



(c)



(d)

Fig. 3. Possible HMMs for modeling duration

3.3 Matching complexity

In the previous section the upper bound on the calculation of functions α and β has been reported. However, adopting the basic block structure suggested in the previous sections for modeling motifs and gaps, this complexity becomes quasi-linear in the number of states. Considering Profile HMMs (see Figure 2), it is immediate to verify that computing α and β has complexity $O(3TN)$, being N the number of states. On the contrary, for both gap models in Figure 3(a) and 3(b) the complexity decreases to $O(2TN)$. Then, the only nonlinear term can be due to the matrix interconnecting the blocks. An experimental evaluation is reported in Figure 4, using a set of S-HMMs of different size (from 160 to 920 states) and sequences of different length (from 633 to 2033 symbols). The number of basic blocks ranges from 6 to 23. Figure 4 reports the Cpu time obtained on a PowerBook G4 for the evaluation of $P(O|\lambda)$ using function α_T . It appears that

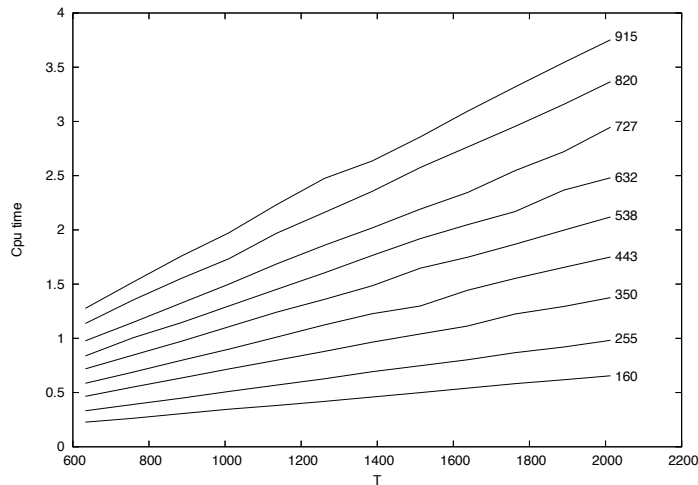


Fig. 4. Complexity for a sequence interpretation task: (a) cpu time versus the string length. Different curves correspond to different number of states.

very complex models can be reasonably used to mine quite long sequences.

4 Modeling Keystroking Dynamics for a Human Agent

The problem of modeling the behavior of an agent may have a very different setting, depending on the kind of agent and on the type of features available for building the model. Here, we will focus on the problem of modeling the behavior of a human agent interacting with a computer system through a keyboard. In

other words, the goal is to construct a model capturing the dynamics of a user typing on a keyboard.

This task has been widely investigated in the past, in order to develop biometric authentication methods (see, for instance [7, 2]), which led to patented solutions [3]. Our purpose is not to provide a new challenging solution to this task, but simply to show how easy it is to generate user models based on S-HMMs, which are performing quite well.

Two case studies have been investigated, characterized by two different targets. The first one addresses the problem of building a discriminant profile of a user, during the activity of typing a free text. This kind of task plays an important role when the goal is to build a monitoring system, which checks on typical behavior of a group of agents and rises an alarm signal when someone of them is not behaving according to its profile.

The second case study aims at producing an authentication procedure based on keystroke dynamics during the login phase, which can be used in order to make more difficult to break into a system in case of password theft.

Notwithstanding the deep difference in the targets, the two experiments share the problem setting, i.e, the input information to match against the behavior model, and the structure of the model.

4.1 Input information

In both cases studies, the input data are sequences of triples $\langle c, t_p, t_r \rangle$, where c denotes the ascii code of the stroked key, t_p the time at which the user begun to press the key, and t_r the time at which the key has been released. The input data are collected by a key-logger transparent to the user.

As the S-HMMs used for building the behavior model rely on the state expansion method for modeling temporal durations, the input data are transformed into symbolic strings, according to the following method: Each keystroke is transformed into a string containing a number of repetitions of the typed character proportional to the duration of the stroke. In a similar way, delay durations have been represented as repetitions of a dummy symbol (".") not belonging to the set of symbols occurring in the typed text. The transformation preserves the original temporal information up to 10 ms accuracy.

4.2 Modeling user behavior

Concerning the user profiling case study, we notice that a *good* user profile should be as much as possible independent from the specific text the user is typing, and, at the same time, should capture its characteristic biometric features. In order to meet these requirements, we constructed a user profile based on specific keywords (denoted as K), such as conjunctive particles, articles, prepositions, auxiliary verbs, and word terminations frequently occurring in any text of a given language. In the specific case, nine K s, from three to four consecutive strokes long, have been selected.

```

SSSSSSS ..... AAAAIIIII ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSS ..... AAAAIIIII ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSSSS ..... AAAAIIIII ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSSSSS ..... IIIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSSSSSDDDD ..... IIIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSSSSS ..... IIIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSSSSS ..... AAAAIIIII ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSS ..... IIIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
IIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSSSS ..... AAAAIIIII ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSSSSSSSS ..... IIIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SS ..... IIIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSS ..... IIIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSDD ..... IIIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
IIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSS ..... IIIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSSSSSS ..... AAAAIIIII ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSS ..... IIIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA
SSSSSSSSSSS ..... IIIIIIAAAAAA ..... TTTT ..... TTTTTT ..... AAAAAAAAAAAAAA

```

Fig. 5. Example of string set obtained by expansion of a word. Typical typing errors are evident, such as the exchange of the A with the I, or double key strokes (S and D pressed simultaneously).

That means that the adherence of a user to its profile is checked only considering the dynamics of the selected key-words as long as they occur during the typing activity. Then, the user profile is reduced to be a collection of nine S-HMM, one for each key-word.

Concerning the user authentication case study, the problem is naturally posed as the one of constructing an S-HMM for each one of the words in the login phase.

However, for both cases some more considerations are necessary. During an editing activity errors due to different factors frequently happen. In many cases, the way in which errors occur is strongly related to the user personality and should be included in the user profile itself. As an example, a very frequent error is the exchange of the order of two keys in the text, when the user types two hands and doesn't have a good synchronization. Another similar situation is when two adjacent keys are simultaneously stroked because of an imperfect perception of the keyboard layout. Example of this kind of typing error are evident in Figure 5. Most mistakes of this kind are self-corrected by the word processors today available, and so do not leave a permanent trace in the final text. Nevertheless, capturing this kind of misbehavior in the user profile greatly improves the robustness of the model.

For the above reason, we made the a-priori choice of modeling substrings corresponding to keystroke expansion by means of basic blocks encoding PHMMs, in order to automatically account for most of the typing errors. Gaps between consecutive keystrokes are simply modeled by a gap model of the kind described in Figure 3-(b).

4.3 Model construction

For every selected word w the corresponding S-HMM modeling the keystroking dynamics of a specific user has been constructed using the algorithm EDY, which has been described in other papers [9, 8]. More specifically, EDY starts with a database LS of learning sequences, and automatically builds an S-HMM λ_{LS} modeling all sequences in LS . In the present case LS contains the strings

encoding the keystroking dynamics for a specific word observed when a specific user was typing (see Figure 5 for an example of string dataset collected for one of the key-words).

A description of EDY and a general evaluation of its performances is outside the scope of this paper. In the following section we will report the performances obtained on the two case studies described above.

5 User Profiling

Ten users collaborated to the experiment by typing a corpus of 2280 words while a key-logger was recording the duration of every stroke and the delay between two consecutive strokes. Each one of the chosen K s occurs in the corpus from 50 to 80 times.

Then, for every user, nine datasets have been constructed, each one collecting the temporal sequences of a specific keyword K .

Let D_{ij} denote the dataset containing the sequences corresponding to keyword K_i typed by the user u_j . Every D_{ij} has been partitioned into a learning set LS_{ij} containing 25 instances and a test set TS_{ij} containing the remaining ones. From every LS_{ij} an S-HMM λ_{ij} has been constructed using the algorithm EDY [9]. EDY was instructed to use PHMM for modeling motifs and the model scheme in Figure 3(b) for modeling gaps.

Then, EDY modeled the strings corresponding to keystrokes as motifs, and the ones corresponding to delays as gaps. It is worth noticing that the mistyping rate was relevant, being key overlapping or key inversion the most frequent mistypes. Consequently, the strings obtained by expansion frequently contained mixtures of two overlapped keys. The S-HMM proved to be an excellent tool for modeling such features, which are highly user specific.

Then, the set $U_j = \{\lambda_{ij} | 1 \leq i \leq 9\}$ of the nine models constructed for every users u_j constitute her profile.

The profile performances have been tested by simulating the editing activity of users typing a text written in the same language as the corpus used to collect the data. The simulation process consisted in generating series of sequences extracted with replacement from the datasets TS_{ij} , which has been compared to the user profile under test.

More precisely, the evaluation procedure was as in the following. Let $\hat{P}(u_j)$ be the probability estimated by means of profile U_j that the observed performance belongs to user u_j . The procedure for testing profile U_j against user u_k ($1 \leq k \leq 10$) is as in the following:

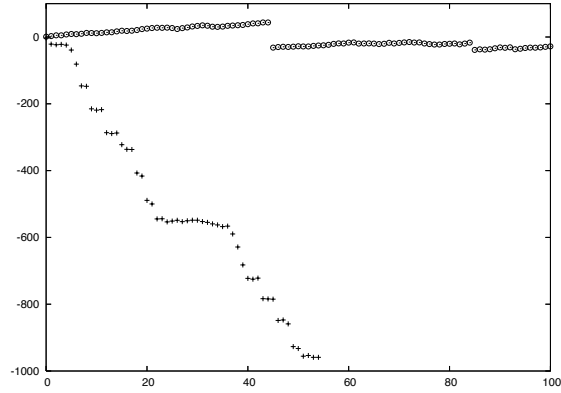
Let initially $\hat{P}(u_j) = 1$.

repeat

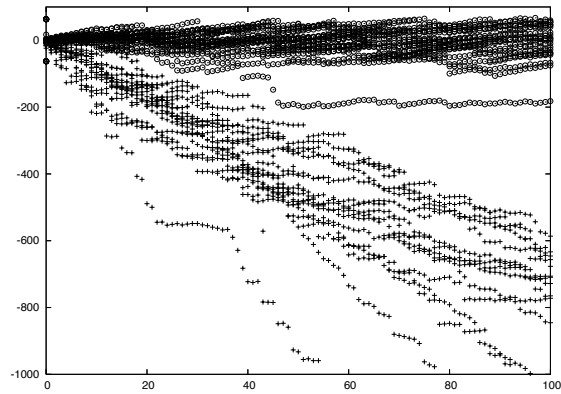
1. Select a sequence O_{ij} from the set TS_{ij} according to the probability distribution of K_i in the corpus.
2. Evaluate the probability $P(O_{ij} | \lambda_{ij})$ that model λ_{ij} has generated O_{ij} .

3. Set $\bar{P}(u_j) = P(u_j) \frac{P(O_{ij}|\lambda_{ij})}{P_{ij}}$.
end

In the above procedure, \bar{P}_{ij} denotes the average probability that model λ_{ij} generates a sequence belonging to its domain, estimated from the corresponding dataset LS_{ij} .



(a)



(b)

Fig. 6. Time evolution, of the logarithm of $\hat{P}(u_j)$. (a) for a single user profile; (b) for all user profiles. Circles describe $\hat{P}(u_j)$ when the performer was u_j . Crosses correspond to $\hat{P}(u_j)$ when the performer was another user.

Figure 6(a) reports the evolution of $\hat{P}(u_j)$ versus the number of keywords progressively found in a text, when $k = j$ and when $k \neq j$. It appears that when the performer corresponds to the profile, $\hat{P}(u_j)$ remains close to 1, whereas it decreases exponentially when the performer does not correspond to the model. Figure 6(b) summarizes in a single diagram the results for all pairing $\langle u_k, U_j \rangle$.

6 User Authentication

As it was impractical to run the experiment using real passwords, it has been supposed that username and password coincide with the pair (name \mathcal{N} , surname \mathcal{S}) of the user. Under this assumption the conjecture that people develop specific skills for typing words strictly related to their own person still holds.

The experiment followed the same protocol described previously. The same group of users typed a number of times their own name and surname, and, then, the name and surname of the other users.

For every user two S-HMMs $\lambda_{\mathcal{N}_j}$, $\lambda_{\mathcal{S}_j}$ have been constructed, for the name and the surname, respectively. The learning sets contained 30 sequences. Figure 7 reports the results of the evaluation of the two models learned for one of the users. For every sequence pair $\langle \mathcal{N}_j, \mathcal{S}_j \rangle$ the probabilities $P(\mathcal{N}_j) = P(O_i | \lambda_{\mathcal{N}_j})$ and $P(\mathcal{S}_j) = P(O_i | \lambda_{\mathcal{S}_j})$ have been evaluated and represented with a point in the plane $\langle P(\mathcal{N}_j), P(\mathcal{S}_j) \rangle$. It is evident that the sequences typed by the user u_j and the ones typed by the other users are separated by a wide margin: in this case, a simple linear discriminator provides a perfect separation. Using a testing set containing 150 negative examples (provided by 9 users different from u_j) and 100 positive examples (provided by the user u_j) for evaluating each one of the models, a discrimination rate of 99% has been obtained. This evaluation does not take into account a small percentage (5%) of positive sequences, which have been rejected because containing abnormal mistakes (for instance, several "backspaces" when the user tried to go back and correct errors he/she noticed). The S-HMM constructed for the name of one of the users is reported in Figure 8. The structure of the model accounts for different typing modalities typical of the user, both concerning the gap duration and the stroke duration. In some cases the model also learned typical mistakes made by the user, such as key inversion or key overlapping. This kind of mistakes happen when the user strike keys with both hands at the same time. The global number of states of the model in Figure 8 is 513. In general, the size of the models constructed for the name or the surname of the considered users ranges from 200 to 600 states, while the response time remains below one second (real-time).

7 Conclusion

In this paper, S-HMMs have been proposed as a general tool for modeling the behavior of an agent, intended both as a human user or as a computer process. The most important property of S-HMM is its ability to model durations by means of the state expansion technique. This is made possible by the low computational cost inherent to S-HMMs, which allows matching in quasi-linear time very complex models.

The ability of S-HMM to accomplish the task of modeling behaviors has been demonstrated on two non trivial case studies of user profiling and of user authentication. The obtained results look very promising, taking also into account the very little effort required to develop the applications.

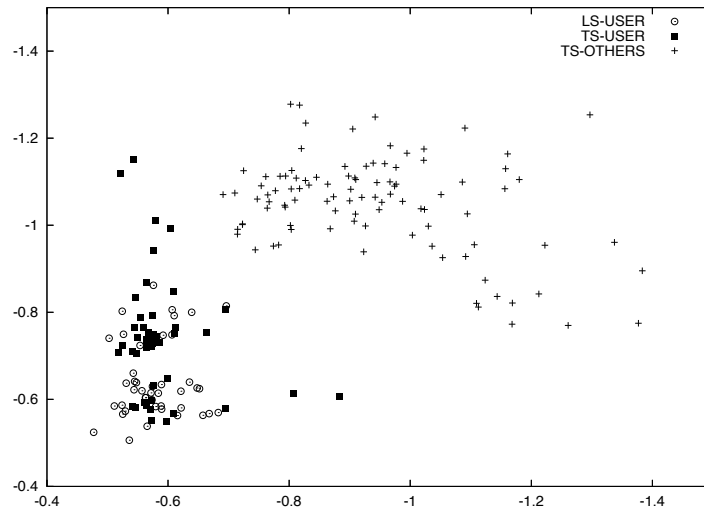


Fig. 7. Results produced by the pair of models learned for the name and the surname of a user. The x axis reports the probability assigned to a sequence by the *name* model. The y axis reports the probability assigned by the *surname* model. Circles denote sequences belonging to the learning set. Black squares denote sequences belonging to the positive testing set, and '+' denotes sequences typed by other users.

References

1. D. Bouchaffra and J. Tan. Structural hidden markov models using a relation of equivalence: Application to automotive designs. *Data Mining and Knowledge Discovery*, 12:7996, 2006.
2. M. Brown and S.J. Rogers. User identification via keystroke characteristics of typed names using neural networks. *International Journal of Man-Machine Studies*, 39:999–1014, 1993.
3. M.E. Brown and S.J. Rogers. Method and apparatus for verification of a computer user's identification, based on keystroke characteristics, patent n. 5,557,686, u.s. patent and trademark office, washington, d.c., Sep 1996.

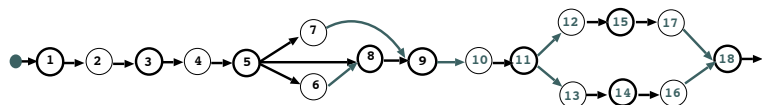


Fig. 8. Example of the S-HMM learned for the name of a user. Circles represent basic blocks encoding models of strokes (thick line) and of gaps between one stroke and another (thin line).

4. Hung Hai Bui, Svetha Venkatesh, and Geoff A. W. West. Tracking and surveillance in wide-area spatial environments using the abstract hidden markov model. *IJPRAI*, 15(1):177–195, 2001.
5. R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis*. Cambridge University Press, 1998.
6. T. Fawcett and F. Provost. Adaptive fraud detection. *Data Mining and Knowledge Discovery Journal*, 1:291–316, 1997.
7. S. Furnell, J. M. Orrissey, P. Sanders, and C. Stockel. Applications of keystroke analysis for improved login security and continuous user authentication. In *Proceedings of the Information and System Security Conference*, pages 283–294, 1996.
8. U. Galassi, M. Botta, and A. Giordana. Hierarchical hidden markov models for user/process profile learning. *Fundamenta Informaticae*, 78:1–19, 2007.
9. U. Galassi, A. Giordana, and L. Saitta. Incremental construction of structured hidden markov models. In *proceedings IJCAI-2007*, pages 2222–2227, 2007.
10. Antonio Bonafonte Josep. Duration modeling with expanded hmm applied to speech recognition.
11. W. Lee and S.J Stolfo. Data mining approaches for intrusion detection. In *Proceedings of the Seventh USENIX Security Symposium (SECURITY '98)*, San Antonio, TX, 1998.
12. W. Lee, w. Fan, M. Miller, S.J. Stolfo, and E. Zadok. Toward cost-sensitive modeling for intrusion detection and response. *Journal of Computer Security*, 10:5 – 22, 2002.
13. S.E. Levinson. Continuous variable duration hidden markov models for automatic speech recognition. *Computer Speech and Language*, 1:29 – 45, 1986.
14. Janne Pykknen and Mikko Kurimo. Using phone durations in finnish large vocabulary continuous speech recognition, 2004.
15. L. Rabiner and B. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, Englewood Cliffs, NY, 1993.
16. L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of IEEE*, 77(2):257–286, 1989.
17. D. Schulz, W. Burgard, D. Fox, and A. Cremens. People tracking with mobile robots using sample-based joint probabilistic data association filters, 2003.
18. S.j. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. Chan. Cost-based modeling for fraud and intrusion detection: Results from the jam project. In *Proceedings of the 2000 DARPA Information Survivability Conference and Exposition (DISCEX '00)*, 2000.
19. S.J. Stolfo, W. Li, S. Hershkop, K. Wang, C. Hu, and O. Nimeskern. Detecting viral propagations using email behavior profiles. *ACM Transactions on Internet Technology (TOIT)*, pages 128–132, 2004.
20. D. Tweed, R. Fisher, J. Bins, and T. List. Efficient hidden semi-markov model inference for structured video sequences. In *Proc. 2nd Joint IEEE Int. Workshop on VSPETS*, pages 247–254, Beijing, China, 2005.
21. S-Z Yu and H. Kobashi. An efficient forward-backward algorithm for an explicit duration hidden markov model. *IEEE Signal Processing Letters*, 10(1), 2003.