Person Movement Prediction Using Hidden Markov Models

Arpad Gellert

Lucian Vintan

Computer Science Department,

"Lucian Blaga" University of Sibiu,

E. Cioran Str., No. 4, Sibiu-550025,

Romania

{arpad.gellert, lucian.vintan}@ulbsibiu.ro

Abstract: Ubiquitous systems use context information to adapt appliance behavior to human needs. Even more convenience is reached if the appliance foresees the user's desires and acts proactively. This paper introduces Hidden Markov Models, in order to anticipate the next movement of some persons. The optimal configuration of the model is determined by evaluating some movement sequences of real persons within an office building. The simulation results show accuracy in next location prediction reaching up to 92%.

Keywords: Ubiquitous Computing, Context Prediction Methods, Hidden Markov Models, Neural Networks

Professor Lucian N. Vinţan, PhD („Lucian Blaga" University of Sibiu, RO) is an active researcher in Advanced Computer Architecture, Context Prediction in Ubiquitous Computing Systems, Parallel and Distributed Systems. He is a member of Academy of Technical Sciences from Romania, European Commission Expert in Computer Science, Visiting Researcher Fellow at University of Hertfordshire, UK. Professor Vinţan published 8 books and over 100 scientific papers (Romania, USA, Italy, UK, Portugal, Hungary, Austria, Germany, Poland, etc.). He introduced some well-known original architectural concepts in Computer Architecture domain (Dynamic Neural Branch Prediction, Pre-Computed Branches, Value Prediction focused on CPU's Context, etc.), recognized, cited and debated through over 50 papers published in many prestigious international conferences and scientific reviews (ACM, IEEE, IEE, etc.)

Arpad Gellert, MSc, is working as a teaching assistant at Computer Science Department from "Lucian Blaga" University of Sibiu, Romania. He received the Master of Science degree in Computer Science at the same university in 2003. Arpad Gellert is currently a PhD student in Computer Science, working on a thesis entitled "*Advanced Prediction Methods Integrated into Speculative Computer Architectures*", having as supervisor professor dr. Lucian Vinţan and co-supervisor professor dr. Theo Ungerer from University of Augsburg, Germany. He is an active researcher in Advanced Computer Architecture, Context Prediction in Ubiquitous Computing Systems and he published over 10 scientific papers in some appreciate reviews (IEE Proc. Computer and Digital Techniques) and international conferences (UK, Germany, Belgium and Romania).

1. Introduction

Ubiquitous systems strive for adaptation to user needs by utilizing information about the current context in which a user's appliance works. A new quality of ubiquitous systems may be reached if context awareness is enhanced by predictions of future contexts based on current and previous context information. Such a prediction enables the system to proactively initiate actions that enhance the convenience of the user or that lead to an improved overall system.

Humans typically act in a certain habitual pattern, however, they sometimes interrupt their behavior pattern and they sometimes completely change the pattern. Our aim is to relieve people of actions that are done habitually without determining a person's action. The system should learn habits automatically and reverse assumptions if a habit changes. The predictor information should therefore be based on previous behavior patterns and applied to speculate on the future behavior of a person. If the speculation fails, the failing must be recognized, and the predictor must be updated to improve future prediction accuracy [4, 5].

For our application domain we chose next location prediction instead of general context prediction. The algorithms may also be applicable for other more general context domains; however, there already exist numerous scenarios within our application's domain. Some sample scenarios may be the following [4]:

- Smart doorplates that are able to direct visitors to the current location of an office owner based on a location-tracking system and predict if the office owner is soon coming back. This scenario is currently developed at Augsburg University and we are involved in the context prediction approach.
- Elevator prediction could anticipate at which floor an elevator will be needed next.
- Routing prediction for cellular phone systems may predict the next radio cell a cellular phone owner will enter based on his previous movement behavior.

To predict or anticipate a future situation, learning techniques as e.g. Markov Chains, Hidden Markov Models, Bayesian Networks, Time Series or Neural Networks are obvious candidates. The challenge is to transfer these algorithms to work with context information.

Petzold, et al., in their work [4], transformed some prediction algorithms used in branch prediction techniques of current high-performance microprocessors, to handle context prediction. They evaluated the one-level one-state, two-state, and multiple-state predictors, and the two-level two-state predictors with local and global first-level histories. The evaluation was performed by simulating the predictors with behavior patterns of people walking through a building as workload. Their simulation results show that the context predictors perform well but exhibit differences in training and retraining speed and in their ability to learn complex patterns.

In another work [5], Petzold, et al., introduced context prediction techniques based on previous behavior patterns, in order to anticipate a person's next movement. They analyzed the two-level predictors with global first-level histories and the two-state predictors and they compared these predictors with the Prediction by Partial Matching (PPM) method. They evaluated the predictors by some movement sequences of real persons within an office building reaching up to 59% accuracy in next location prediction without pre-training and, respectively, up to 98% with pre-training.

Rabiner in his work [7], shows how HMMs can be applied to selected problems in speech recognition. His paper presents the theory of HMMs from the simplest concepts (discrete Markov chains) to the most sophisticated models (variable duration, continuous density models, etc.). He also illustrated some applications of the theory of HMMs to simple problems in speech recognition, and pointed out how the techniques have been applied to more advanced speech recognition problems.

Liu et al. in their work [3], describe a HMM based framework for hand gesture detection and recognition. The goal of gesture interpretation is to improve human-machine communication and to bring human-machine interaction closer to human-human interaction, making possible new applications such as sign language translation. They present an efficient method for extracting the observation sequence using the feature model and Vector Quantization, and demonstrate that, compared to the classic template-based methods, the HMM-based approach offers a more flexible framework for recognition.

Galata et al. in their work [2], present a novel approach for automatically acquiring stochastic models of the high-level structure of a human activity without the assumption of any prior knowledge. The process involves temporal segmentation into plausible atomic behaviour components and the use of variable length Markov models for the efficient representation of behaviours. Their experimental results demonstrate that the use of variable length Markov models provides an efficient mechanism for learning complex behavioral dependencies and constraints.

Machine Learning techniques based on HMMs has been also applied to problems in computational biology and they can be used as mathematical models of molecular processes and biological sequences. The goal of computational biology is to elucidate additional information required for drug design, medical diagnosis and medical treatment. The majority of molecular data used in computational biology consists in sequences of nucleotides corresponding to the primary structure of DNA and RNA, or sequences of amino acids corresponding to the primary structure of proteins. Birney in his work [1], reviews gene-prediction HMMs and protein family HMMs. The role of gene-prediction in DNA is to discover the location of genes on the genome. HMMs have also been used in protein profiling to discriminate between different protein families and predict a new protein-family or subfamily. Yoon et al. in their work [10], proposed a new method based on context-sensitive HMMs, which can be used for predicting RNA secondary structure. The RNA secondary structure results from the base pairs formed by the nucleotides of RNA. The context-sensitive HMM can be viewed as an extension of the traditional HMM, where some of the states are equipped with auxiliary memory. Symbols that are emitted at certain states are stored in the memory, and they serve as the context that affects the emission and transition probabilities of the model. They demonstrated that the proposed model predicts the secondary structure very accurately, at a low computational cost.

This paper focuses on a Hidden Markov Model (HMM) approach, introducing the HMM-based predictors and comparing them with simple Markov and respectively neural predictors. Our application predicts the next room based on the history of rooms, visited by a certain person moving within an office building. We evaluate these predictors by some movement sequences of real persons, acquired from the Smart Doorplates project developed at Augsburg University [4, 5, 6]. The next sections describe the proposed Hidden Markov Models and present the simulation results.

2. Hidden Markov Models of order 1

2.1. Elements of a HMM of Order 1

1. N - the number of hidden states, with S = {S_0, S_1, …, S_{N-1}} the set of hidden states, and q_t the hidden state at time *t*. N will be varied in order to obtain the optimal value.

2. M - the number of observable states, with V = {V_0, V_1, …, V_{M-1}} the set of observable states (symbols), and O_t the observable state at time *t*.

3. A = {a_{ij}} - the transition probabilities between the hidden states S_i and S_j, where $a_{ij} = P[q_{t+1} = S_j \,|\, q_t = S_i], \quad 0 \le i, j \le N-1$.

4. B = {$b_j(k)$} - the probabilities of the observable states V_k in hidden states S_j, where $b_j(k) = P[O_t = V_k \,|\, q_t = S_j], \quad 0 \le j \le N-1, \quad 0 \le k \le M-1$.

5. π = {π_i} - the initial hidden state probabilities, where $\pi_i = P[q_1 = S_i], \quad 0 \le i \le N-1$.

There are also defined the following variables:

- $\alpha_t(i) = P(O_1 O_2 \ldots O_t, q_t = S_i \,|\, \lambda)$ - the forward variable [7], representing the probability of the partial observation sequence until time *t*, and hidden state S_i at time *t*, given the model $\lambda = (A, B, \pi)$.
- $\beta_t(i) = P(O_{t+1} O_{t+2} \ldots O_T \,|\, q_t = S_i, \lambda)$ - the backward variable [7], representing the probability of the partial observation sequence from *t+1* to the end *T*, given hidden state S_i at time *t* and the model $\lambda = (A, B, \pi)$.
- $\xi_t(i, j) = P(q_t = S_i, q_{t+1} = S_j \,|\, O_1 O_2 \ldots O_T, \lambda)$ - the probability of being in hidden state S_i at time *t*, and hidden state S_j at time *t+1*, given the model $\lambda = (A, B, \pi)$ and the observation sequence.
- $\gamma_t(i) = P(q_t = S_i \,|\, O_1 O_2 \ldots O_T, \lambda)$ - the probability of being in hidden state S_i at time *t*, given the model $\lambda = (A, B, \pi)$ and the observation sequence.
- H - the history (the number of observations used in the prediction process). In [7] and [8] the entire observation sequence is used in the prediction process (H=T), but in some practical applications the observation sequence increases continuously, therefore it is necessary its limitation. Thus, the last *H* observations can be stored in a left shift register.
- I - the maximum number of iterations in the adjustment process. Usually the adjustment process ends when the probability of the observation sequence doesn't increase anymore, but for a faster adjustment, it is limited the number of iterations.

2.2. Adjustment Process of a HMM of Order 1

1. **INITIALIZE** $\lambda = (A, B, \pi)$;
2. **COMPUTE** $\alpha_t(i), \beta_t(i), \xi_t(i, j), \gamma_t(i), \quad t = 1, \ldots, T, \quad i = 0, \ldots, N-1, \quad j = 0, \ldots, N-1$;
3. **ADJUST THE MODEL** $\lambda = (A, B, \pi)$;
4. **IF** $P(O \,|\, \lambda)$ **INCREASES, GO TO 2.**

2.3. Initialization of the Model of Order 1

- The transition probabilities between the hidden states A(NXN) = {aij}, are randomly initialized to approximately 1/N, each row summing to 1.
- The probabilities of the observable states B(NXM) = {bj(k)}, are randomly initialized to approximately 1/M, each row summing to 1.
- The initial hidden state probabilities π(1XN) = {πi} are randomly set to approximately 1/N, their sum being 1.

2.4. Prediction Algorithm Using a HMM of Order 1

1. T=1 (T is the length of the observation sequence);
2. T=T+1;

   if T<H go to 2.)
3. c=0 (c is the number of current iteration, its maximum value is given by I);
4. The model $\lambda = (A, B, \pi)$ is repeatedly adjusted based on the last H observations $O_{T-H+1}, O_{T-H+2}, ..., O_T$ (the entire observation sequence if H=T), in order to increase the probability of the observation sequence $P(O_{T-H+1}\, O_{T-H+2} \ldots O_T | \lambda)$. In 4.1, 4.2 and 4.3 steps the denominators are used in order to obtain a probability measure, and to avoid underflow. As Stamp showed in [8], underflow is inevitable without scaling, since the probabilities tend to 0 exponentially as T increases.

   4.1. Compute the forward variable α in a recursive manner:

   $$\alpha_{T-H+1}(i) = \frac{\pi_i \cdot b_i(O_{T-H+1})}{\sum_{i=0}^{N-1} \pi_i \cdot b_i(O_{T-H+1})},\ i = 0,..., N-1,$$ where $\alpha_{T-H+1}(i)$ is the probability of observation symbol O_{T-H+1} and initial hidden state S_i, given the model $\lambda = (A, B, \pi)$;

   $$\alpha_t(j) = \frac{\sum_{i=0}^{N-1} \alpha_{t-1}(i) \cdot a_{ij} \cdot b_j(O_t)}{\sum_{j=0}^{N-1}\sum_{i=0}^{N-1} \alpha_{t-1}(i) \cdot a_{ij} \cdot b_j(O_t)},\ t = T-H+2,...,T,\ \ j = 0,..., N-1,$$ where

   $\alpha_t(j)$ is the probability of the partial observation sequence until time t ($O_{T-H+1} \ldots O_t$), and hidden state S_j at time t, given the model $\lambda = (A, B, \pi)$. Since, by definition,

   $$\alpha_T(j) = P(O_{T-H+1}\, O_{T-H+2} \ldots O_T,\ q_T = S_j | \lambda),$$

   the sum of the terminal forward variables $\alpha_T(j)$ gives the probability of the observation sequence:

   $$P(O_{T-H+1}\, O_{T-H+2} \ldots O_T | \lambda) = \sum_{j=0}^{N-1} \alpha_T(j).$$

   4.2. Compute the backward variable β in a recursive manner:

   $$\beta_T(i) = \frac{1}{\sum_{j=0}^{N-1}\sum_{i=0}^{N-1} \alpha_{T-1}(i) \cdot a_{ij} \cdot b_j(O_T)},\ i = 0,... N-1;$$

   $$\beta_t(i) = \frac{\sum_{j=0}^{N-1} a_{ij} \cdot b_j(O_{t+1}) \cdot \beta_{t+1}(j)}{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} a_{ij} \cdot b_j(O_{t+1}) \cdot \beta_{t+1}(j)},\quad t = T-1,..., T-H+1,\quad i = 0,..., N-1,$$

   where

   $\beta_t(i)$ is the probability of the partial observation sequence from $t+1$ to the end T ($O_{t+1}O_{t+2}\ldots O_T$), given hidden state S_i at time t and the model $\lambda = (A, B, \pi)$.

   4.3. Compute ξ:

$$\xi_t(i,j)=\frac{\alpha_t(i)\cdot a_{ij}\cdot b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\alpha_t(i)\cdot a_{ij}\cdot b_j(O_{t+1})\beta_{t+1}(j)},\quad t=T-H+1,...,T-1,\quad i=0,...,N-1,$$

$$j=0,...,N-1$$

where $\xi_t(i,j)$ is the probability of being in hidden state S_i at time t and respectively S_j at time $t+1$, given the observation sequence $O_{T-H+1}\,O_{T-H+2}\,...\,O_T$ and the model $\lambda=(A,B,\pi)$.

4.4. Compute γ:

$\gamma_t(i)=\sum_{j=0}^{N-1}\xi_t(i,j),\ \ t=T-H+1,...,T-1,\ \ i=0,...,N-1,$ where $\gamma_t(i)$ is the probability of being in the hidden state S_i at time t, given the model $\lambda=(A,B,\pi)$ and the observation sequence $O_{T-H+1}\,O_{T-H+2}\,...\,O_T$.

4.5. Adjust π:

$\overline{\pi_i}=\gamma_{T-H+1}(i)$ - represents the expected number of times the hidden state is S_i at the initial time $t=T-H+1$.

4.6. Adjust A:

$\overline{a_{ij}}=\frac{\sum_{t=T-H+1}^{T-1}\xi_t(i,j)}{\sum_{t=T-H+1}^{T-1}\gamma_t(i)}$ - represents the probability of transition from hidden state S_i to S_j.

The numerator is the expected number of transitions from state S_i to S_j, while the denominator is the expected number of transitions from state S_i to any state.

4.7. Adjust B:

$\overline{b_j(k)}=\frac{\sum_{t=T-H+1,\ O_t=V_k}^{T-1}\gamma_t(j)}{\sum_{t=T-H+1}^{T-1}\gamma_t(j)}$ - the probability of observation symbol V_k given that the model is in hidden state S_j. The numerator is the expected number of times the model is in hidden state S_j and the observation symbol is V_k, while the denominator is the expected number of times the model is in hidden state S_j.

4.8. c=c+1;

if $\log[P(O_{T-H+1}...O_T|\overline{\lambda})] > \log[P(O_{T-H+1}...O_T|\lambda)]$ and c<I then go to 4.).

Since P would be out of the dynamic range of the machine [7], we compute the log of P, using the following formula [8]:

$$\log[P(O_{T-H+1}\,...\,O_T|\overline{\lambda})]=-\log\left(\frac{1}{\sum_{i=0}^{N-1}\pi_i\cdot b_i(O_{T-H+1})}\right)-\sum_{t=T-H+2}^{T}\log\left(\frac{1}{\sum_{j=0}^{N-1}\sum_{i=0}^{N-1}\alpha_{t-1}(i)\cdot a_{ij}\cdot b_j(O_t)}\right)$$

1.) At current time T, it is predicted the next observation symbol O_{T+1}, using the adjusted model $\overline{\lambda}=(\overline{A},\overline{B},\overline{\pi})$:

- choose hidden state S_i at time T, i=0, …, N-1, maximizing $\alpha_T(i)$;
- choose next hidden state S_j (at time T+1), j=0, …, N-1, maximizing $\overline{a_{ij}}$;
- predict next symbol V_k (at time T+1), k=0, …, M-1, maximizing $\overline{b_j(k)}$.

If the process continues, then T=T+1 and go to 3.).

3. A Possible Generalization: Hidden Markov Models of Order R

In this paragraph we try to develop a Hidden Markov Model of order R, R≥1. There are multiple possibilities for doing this but we present here only one we considered the most appropriate. The key of our proposed model is represented by the so-called hidden super-states, a combination of R primitive hidden states. Therefore, the main difference, comparing with an order 1 HMM, consists in the fact that the stochastic hidden Markov model is of order R instead of order one. This new model is justified because we suppose that in some specific applications, there are longer correlations within the hidden state model. In other words, we suppose that the next hidden state is better determined by the current super-state rather than by the current primitive state. As it can be further seen, the new proposed model is similar with the well-known HMM of order one, excepting that the generic primitive hidden state becomes now a generic super-state.

3.1. Elements of a HMM of Order R

1. R - the order of HMM (a combination of R primitive hidden states form a so called super-state).
2. N - the number of primitive hidden states (belonging to a HMM of order 1), with $S = \{S_0, S_1, ..., S_{N^R-1}\}$ being the set of hidden super-states and $q_t \in S$ the hidden super-state at time *t*. The current super-state determines the transition into the next one based on a super-state transition matrix with restrictions (this transition matrix involve a non-ergodic model, see example of Table 1). N will be varied in order to obtain the optimal value.
3. M - the number of observable states, with V = {V_0, V_1, …, V_{M-1}} the set of observable states (symbols), and O_t the observable state at time *t*.
4. A = {a_{ij}} - the transition probabilities between the hidden super-states S_i and S_j, where $a_{ij} = P[q_{t+1} = S_j | q_t = S_i], \quad 0 \le i, j \le N^R - 1$.
5. B = {$b_j(k)$} - the probabilities of the observable states V_k, considering the current hidden super-state S_j, where $b_j(k) = P[O_t = V_k | q_t = S_j], \quad 0 \le j \le N^R - 1, \quad 0 \le k \le M - 1$.
6. π = {π_i} - the initial hidden super-state probabilities, where $\pi_i = P[q_1 = S_i], \quad 0 \le i \le N^R - 1$.

In order to simplify the terminology, in the rest of the paper we'll refer to the hidden super-states as simply hidden states belonging to the HMM of order R.

We also define the following variables:

- $\alpha_t(i) = P(O_1 O_2 ... O_t, q_t = S_i | \lambda)$ - the forward variable [7], representing the probability of the partial observation sequence until time *t*, and hidden state S_i at time *t*, given the model $\lambda = (A, B, \pi)$.
- $\beta_t(i) = P(O_{t+1} O_{t+2} ... O_T | q_t = S_i, \lambda)$ - the backward variable [7], representing the probability of the partial observation sequence from *t+1* to the end *T*, given hidden state S_i at time *t* and the model $\lambda = (A, B, \pi)$.
- $\xi_t(i, j) = P(q_t = S_i, q_{t+1} = S_j | O_1 O_2 ... O_T, \lambda)$ - the probability of being in hidden state S_i at time *t*, and hidden state S_j at time *t+1*, given the model $\lambda = (A, B, \pi)$ and the observation sequence.
- $\gamma_t(i) = P(q_t = S_i | O_1 O_2 ... O_T, \lambda)$ - the probability of being in hidden state S_i at time *t*, given the model $\lambda = (A, B, \pi)$ and the observation sequence.
- H - the history (the number of observations used in the prediction process). In [7] and [8] the entire observation sequence is used in the prediction process (H=T), but in some practical applications the observation sequence increases continuously, therefore it is necessary its limitation. Thus, the last *H* observations can be stored in a left shift register having a certain length.

– I - the maximum number of iterations in the adjustment process. Usually the adjustment process ends when the probability of the last H observations doesn't increase anymore, but for a faster adjustment, it is limited the number of iterations.

For a HMM of order R with N hidden states, the transition probabilities between the hidden states $A(N^R x N^R) = \{a_{ij}\}$, are stored in a table with N^R rows and N^R columns but not all cells of the table are used; there are only N consistent (possible) transitions from each state involving a non-ergodic model. The following table, for example, corresponds to a HMM of order 3 (R=3) with 2 primitive hidden states (N=2):

			j							
			0	1	2	3	4	5	6	7
			AAA	***AAB***	ABA	ABB	BAA	BAB	BBA	BBB
i	0	AAA	X	X						
	1	AAB			X	X				
	2	ABA					X	X		
	3	ABB							X	X
	4	BAA	X	X						
	5	BAB			X	X				
	6	BBA					X	X		
	7	BBB							X	X

Table 1. Consistent Transitions for a HMM of Order 3 (R=3), with 2 Hidden States (N=2)

There are used only the consistent cells marked with "X", because transitions are possible only between states which end and respectively start with the same (R-1) primitive hidden states. The consistent cells of the transition table are given by the following formulas:

– For next hidden states (columns) $j = 0,...,N^R - 1$, are consistent only the current hidden states (rows) $i = \left[\frac{j}{N}\right] + 0 \cdot N^{R-1},...,\left[\frac{j}{N}\right] + (N-1) \cdot N^{R-1}$;

– For current hidden states (rows) $i = 0,...,N^R - 1$, are consistent only the next hidden states (columns) $j = (i \bmod N^{R-1}) \cdot N,...,(i \bmod N^{R-1}) \cdot N + N - 1$.

3.2. Adjustment Process of a HMM of Order R

1. Initialize $\lambda = (A, B, \pi)$;
2. Compute $\alpha_t(i),\ \beta_t(i),\ \xi_t(i,j),\ \gamma_t(i),\ \ t = 1,...,T,\quad i = 0,...,N^R - 1,\quad j = 0,...,N^R - 1$;
3. Adjust the model $\lambda = (A, B, \pi)$;
4. If $P(O|\lambda)$ increases, go to 2.

3.3. Initialization of the Model of Order R

1. The transition probabilities between the hidden states $A(N^R X N^R) = \{a_{ij}\}$, are randomly initialized to approximately 1/N; the sum of each row's elements must be 1. The hidden state transition probabilities are initialized for $i = 0,...,N^R - 1$ and $j = (i \bmod N^{R-1}) \cdot N,...,(i \bmod N^{R-1}) \cdot N + N - 1$.
2. The probabilities of the observable states $B(N^R XM) = \{b_j(k)\}$, are randomly initialized to approximately 1/M; the sum of each row's elements must be 1.
3. The initial hidden state probabilities $\pi(1XN^R) = \{\pi_i\}$ are randomly set to approximately $1/N^R$, their sum being 1.

3.4. Prediction Algorithm Using a HMM of Order R

1. T=1 (T is the length of the observation sequence);
2. T=T+1;

   if T<H go to 2.)
3. c=0 (c is the number of current iteration, its maximum value is given by I);
4. The model $\lambda = (A, B, \pi)$ is repeatedly adjusted based on the last H observations $O_{T-H+1}, O_{T-H+2}, ..., O_T$ (the entire observation sequence if H=T), in order to increase the probability of the observation sequence $P(O_{T-H+1}\, O_{T-H+2} \ldots O_T | \lambda)$. In 4.1, 4.2 and 4.3 the denominators are used in order to obtain a probability measure, and to avoid underflow. As Stamp showed in [8], underflow is inevitable without scaling, since the probabilities tend to 0 exponentially as T increases.

   4.1. Compute the forward variable α in a recursive manner:

   $$\alpha_{T-H+1}(i) = \frac{\pi_i \cdot b_i(O_{T-H+1})}{\sum_{i=0}^{N^R-1} \pi_i \cdot b_i(O_{T-H+1})},\ i = 0,..., N^R - 1,$$ where $\alpha_{T-H+1}(i)$ is the probability of observation symbol O_{T-H+1} and initial hidden state S_i, given the model $\lambda = (A, B, \pi)$;

   $$\alpha_t(j) = \frac{\sum_{i=\left[\frac{j}{N}\right]+0\cdot N^{R-1}}^{\left[\frac{j}{N}\right]+(N-1)\cdot N^{R-1}} \alpha_{t-1}(i) \cdot a_{ij} \cdot b_j(O_t)}{\sum_{j=0}^{N^R-1} \sum_{i=\left[\frac{j}{N}\right]+0\cdot N^{R-1}}^{\left[\frac{j}{N}\right]+(N-1)\cdot N^{R-1}} \alpha_{t-1}(i) \cdot a_{ij} \cdot b_j(O_t)},\quad t = T-H+2,...,T,\quad j = 0,..., N^R - 1,$$

   where $\alpha_t(j)$ is the probability of the partial observation sequence until time t ($O_{T-H+1} \ldots O_t$), and hidden state S_j at time t, given the model $\lambda = (A, B, \pi)$. Since, by definition,

   $$\alpha_T(j) = P(O_{T-H+1}\, O_{T-H+2} \ldots O_T,\ q_T = S_j | \lambda),$$

   the sum of the terminal forward variables $\alpha_T(j)$ gives the probability of the observation sequence:

   $$P(O_{T-H+1}\, O_{T-H+2} \ldots O_T | \lambda) = \sum_{j=0}^{N^R-1} \alpha_T(j).$$

   4.2. Compute the backward variable β in a recursive manner:

   $$\beta_T(i) = \frac{1}{\sum_{j=0}^{N^R-1} \sum_{i=\left[\frac{j}{N}\right]+0\cdot N^{R-1}}^{\left[\frac{j}{N}\right]+(N-1)\cdot N^{R-1}} \alpha_{T-1}(i) \cdot a_{ij} \cdot b_j(O_T)},\ i = 0,... N^R - 1;$$

   $$\beta_t(i) = \frac{\sum_{j=(i \bmod N^{R-1})\cdot N}^{(i \bmod N^{R-1})\cdot N+N-1} a_{ij} \cdot b_j(O_{t+1}) \cdot \beta_{t+1}(j)}{\sum_{i=0}^{N^R-1} \sum_{j=(i \bmod N^{R-1})\cdot N}^{(i \bmod N^{R-1})\cdot N+N-1} a_{ij} \cdot b_j(O_{t+1}) \cdot \beta_{t+1}(j)},\ t = T-1,...,T-H+1,\ i = 0,..., N^R - 1,$$

where $\beta_t(i)$ is the probability of the partial observation sequence from *t+1* to the end *T* ($O_{t+1}O_{t+2}\ldots O_T$), given hidden state S_i at time *t* and the model $\lambda = (A, B, \pi)$.

4.3. Compute ξ:

$$\xi_t(i,j) = \frac{\alpha_t(i) \cdot a_{ij} \cdot b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=0}^{N^R-1} \sum_{j=(i \bmod N^{R-1})\cdot N}^{(i \bmod N^{R-1})\cdot N+N-1} \alpha_t(i) \cdot a_{ij} \cdot b_j(O_{t+1})\beta_{t+1}(j)}, \quad t = T-H+1,\ldots,T-1,$$

$i = 0,\ldots,N^R-1, j = (i \bmod N^{R-1}) \cdot N,\ldots,(i \bmod N^{R-1}) \cdot N + N - 1,$ where $\xi_t(i,j)$ is the probability of being in hidden state S_i at time *t* and respectively S_j at time *t+1*, given the observation sequence $O_{T-H+1}\, O_{T-H+2} \ldots O_T$ and the model $\lambda = (A, B, \pi)$.

4.4. Compute γ:

$\gamma_t(i) = \sum_{j=(i \bmod N^{R-1})\cdot N}^{(i \bmod N^{R-1})\cdot N+N-1} \xi_t(i,j), \quad t = T-H+1,\ldots,T-1, \quad i = 0,\ldots,N^R-1$, where $\gamma_t(i)$ is the probability of being in hidden state S_i at time *t*, given the model $\lambda = (A, B, \pi)$ and the observation sequence $O_{T-H+1}\, O_{T-H+2} \ldots O_T$.

4.5. Adjust π:

$\overline{\pi_i} = \gamma_{T-H+1}(i)$ - represents the expected number of times the hidden state is S_i $(i = 0,\ldots,N^R-1)$ at the initial time $t = T-H+1$.

4.6. Adjust *A*:

$\overline{a_{ij}} = \frac{\sum_{t=T-H+1}^{T-1} \xi_t(i,j)}{\sum_{t=T-H+1}^{T-1} \gamma_t(i)}$ - the probability of transition from hidden state S_i to S_j, where $i = 0,\ldots,N^R-1$ and $j = (i \bmod N^{R-1}) \cdot N,\ldots,(i \bmod N^{R-1}) \cdot N + N - 1$. The numerator is the expected number of transitions from state S_i to S_j, while the denominator is the expected number of transitions from state S_i to any state.

4.7. Adjust *B*:

$\overline{b_j(k)} = \frac{\sum_{\substack{t=T-H+1 \\ O_t=V_k}}^{T-1} \gamma_t(j)}{\sum_{t=T-H+1}^{T-1} \gamma_t(j)}$ - the probability of observation symbol V_k ($k = 0,\ldots,M-1$) given that the model is in hidden state S_j ($j = 0,\ldots,N^R-1$). The numerator is the expected number of times the model is in hidden state S_j and the observation symbol is V_k, while the denominator is the expected number of times the model is in hidden state S_j.

4.8. c=c+1;

if $\log[P(O_{T-H+1}\ldots O_T|\overline{\lambda})] > \log[P(O_{T-H+1}\ldots O_T|\lambda)]$ and c<I then go to 4.).

Since *P* would be out of the dynamic range of the machine [7], we compute the log of *P*, using the following formula [8]:

$$\log[P(O_{T-H+1}\ldots O_T|\overline{\lambda})] = -\log\left(\frac{1}{\sum_{i=0}^{N^R-1}\pi_i\cdot b_i(O_{T-H+1})}\right) - \sum_{t=T-H+2}^{T}\log\left(\frac{1}{\sum_{j=0}^{N^R-1}\sum_{i=\left[\frac{j}{N}\right]+0\cdot N^{R-1}}^{\left[\frac{j}{N}\right]+(N-1)\cdot N^{R-1}}\alpha_{t-1}(i)\cdot a_{ij}\cdot b_j(O_t)}\right)$$

1.) At time T, it is predicted the next observation symbol O_{T+1}, using the adjusted model $\overline{\lambda} = (\overline{A}, \overline{B}, \overline{\pi})$:

- choose hidden state S_i at time T, $i = 0,\ldots,N^R - 1$, maximizing $\alpha_T(i)$;
- choose next hidden state S_j (at time T+1), $j = (i \bmod N^{R-1})\cdot N,\ldots,(i \bmod N^{R-1})\cdot N + N - 1$, maximizing $\overline{a_{ij}}$;
- predict next symbol V_k (at time T+1), $k = 0,\ldots,M-1$, maximizing $\overline{b_j(k)}$.

If the process continues, then T=T+1 and go to 3.).

4. Experimental Results

Our application predicts the next room based on the history of rooms, visited by a certain person moving within an office building. We evaluate these HMM predictors by some movement sequences of real persons developed by the research group at the University of Augsburg [6]. In this work we are interested in predicting the next room from all rooms except for the own office.

Each line from the original benchmarks [6] represents a person's movement (his/her entry in a room). It contains the movement's date and hour, the room's name, the person's name and a timestamp. In the codification process we eliminated from the benchmark the common corridor, because it could behave as noise. Table 2 shows how looks the benchmark before and after the room codification process.

Original benchmark	**Benchmark after room codification**
2003.07.07 10:13:45; 402; Employee2; 1057565625801	0
2003.07.07 10:21:41; corridor; Employee2; 1057566101067	-
2003.07.07 10:21:45; 411; Employee2; 1057566105152	1
2003.07.07 10:21:48; corridor; Employee2; 1057566108771	-
2003.07.07 10:21:54; 402; Employee2; 1057566114338	0

Table 2. The First Lines From a Certain Benchmark (With a Movement Sequence of Employee 2) Before and After the Room Codification Process.

After the codification process the benchmarks contain only the room codes (0÷13), because in this starting stage of our work only this information is used in the prediction process. We used two benchmark types: some short benchmarks containing about 300-400 movements and some long benchmarks containing about 1000 movements. We used the short benchmarks to pre-train the HMM predictors, and the long benchmarks for evaluations.

We started with a HMM of order 1 with 2 hidden states (N=2), and we tested it on the fall benchmarks – developed at Augsburg University [6] – without corridor. The following simulation methodology was used in order to predict each room from a certain benchmark: we predicted the next room at time t based on the entire room sequence from that benchmark until time t-1. We compared a HMM without attached confidence automata with HMMs using different confidence automata. We denoted as *n rooms m states conf* an *m*-state confidence counter, associated to each sequence of the last *n* rooms visited by the person. The 4-state automata have 2 predictable states, while the 2-state automata have only 1 predictable state. In the case of using 4-state automata, a prediction is generated in each of the two predictable states [9]. Table 3 shows that the best average prediction accuracy (AM) is obtained when using a 4-state confidence for each sequence of 2 rooms:

Benchmark	no conf	2 rooms 4 states conf	2 rooms 2 states conf	1 room 4 states conf
Employee 1	91.89	92.68	**90.80**	93.33
Employee 2	77.9	84.06	86.41	79.24
Employee 3	65.66	73.82	71.05	69.35
Boss	79.07	84.72	84.07	84.35
AM	78.63	83.82	83.0825	81.5675

Table 3. Comparing a HMM Without Confidence with HMMs Using Different Types of Confidence.

We continued our simulations varying the number of hidden states. We used again the fall benchmarks [6], and a HMM of order 1 with 4-state confidence automata associated to each sequence of two rooms. Table 4 shows how is affected the prediction accuracy of a HMM by the number of hidden states:

Benchmark	N=1	N=2	N=3	N=4	N=5	N=6	N=7
Employee 1	92.68	92.68	92.68	92.68	92.68	92.68	92.68
Employee 2	84.15	84.06	84	84.57	**82.84**	82.45	83.13
Employee 3	73.37	73.82	74.65	74.63	76.55	73.91	74.04
Boss	84.72	84.72	85.6	85.49	87.17	85.82	85.6
AM	83.73	83.82	84.2325	84.3425	**84.81**	83.715	83.8625

Table 4. Study of the Number of Hidden States Using HMM with 4-State Confidence Automata.

It can be observed in Table 4 that for a HMM of order 1, the optimal number of hidden states is 5. Our scientific hypothesis is that the hidden states are the five working days of a week. We varied again the number of hidden states without using the confidence automata. Table 5 shows that in this case the optimal number of hidden states is 1.

Benchmark	N=1	N=2	N=3	N=4	N=5	N=6	N=7
Employee 1	91.89	91.89	91.89	91.89	91.89	91.89	90.09
Employee 2	79.77	77.9	76.77	76.02	75.65	74.53	75.65
Employee 3	70.18	65.66	64.9	64.15	65.28	61.13	64.9
Boss	79.49	79.07	72.38	71.12	72.8	71.54	71.12
AM	**80.33**	78.63	76.48	75.79	76.4	74.77	75.44

Table 5. Study of the Number of Hidden States Using HMM Without Confidence Automata.

We compare now the best HMM of order 1 with different configurations of some "equivalent" NNs (Neural Networks) developed in our previous work [9] and respectively simple Markov predictors. "Equivalent" means in this case that we compare all these different predictors considering that they have, however, the same inputs. We used the HMM of order 1 with 5 hidden states considering an attached 4-state confidence automata. We compared it with a statically pre-trained NN with a room history length of 1 (NN of order 1) and 2 (NN of order 2), a learning rate of 0.2 and a threshold of 0.3 (having the same confidence automata). We also compared the HMM with some simple Markov predictors of order 1 and 2 (having the same confidence automata). The NN was statically trained on the summer benchmarks [6], and each predictor was tested on the longer fall benchmarks [6]. Table 6 presents the prediction accuracies obtained with the HMM, NN and Markov predictors, all with 4-state confidence automata associated to each sequence of two rooms:

Benchmark	HMM of order 1 (N=5)	NN of order 1	NN of order 2	Markov of order 1	Markov of order 2
Employee 1	92.68	92.4	92.5	93.42	93.33
Employee 2	82.84	84.35	85.96	**83.43**	84.33
Employee 3	76.55	73.91	74.17	73.75	75
Boss	87.17	86.71	85.93	85.82	85.71
AM	**84.81**	84.34	84.64	84.1	84.59

Table 6. Comparing a HMM of Order 1 with NN and Markov Predictors Using 4-State Confidence Automata.

Employee 2 hasn't got a behavior correlated with days of week (see Table 4 and also Table 6). It can be seen in Table 6 too, because the Markov predictor of order 1 outperforms the HMM of order 1. NNs are better than HMM on Employee 2, but we suspect that these results are too optimistic due to the NN's pre-training (first case) and respectively due to the NN's order 2 (second case).

Table 7 presents the prediction accuracies obtained with the HMM, NN and simple Markov predictors (the same configuration) without confidence automata:

Benchmark	HMM of order 1 (N=1)	NN of order 1	NN of order 2	Markov of order 1	Markov of order 2
Employee 1	91.89	89.18	90.09	81.98	75.67
Employee 2	79.77	77.15	76.4	74.15	70.03
Employee 3	70.18	62.26	67.54	64.9	56.98
Boss	79.49	76.56	75.73	73.64	62.34
AM	**80.33**	76.28	77.44	73.66	66.25

Table 7. Comparing a HMM of order 1 with NN and simple Markov predictors without confidence automata.

In order to confirm or infirm our scientific hypothesis that, in the case of HMM with 4-state confidence automata, the five hidden states might be the working days of the week, we implemented a predictor which consists of five simple Markov predictors. Each Markov predictor is associated to one of the 5 days (Monday, Tuesday, …, Friday). Table 8 compares this predictor containing 5 simple Markov predictors with HMMs and simple Markov predictors, using the 4-state confidence automata, and shows that only on one person (Boss) is confirmed our hypothesis.

Benchmark	HMM of order 1 (N=5)	Markov of order 1	5 Markov predictors of order 1
Employee 1	92.68	93.42	86.53
Employee 2	82.84	83.43	78.44
Employee 3	76.55	73.75	55.81
Boss	87.17	85.82	**88.46**
AM	84.81	84.1	77.31

Table 8. Comparing a HMM of order 1 with Markov of order 1 and a predictor consisting in 5 Markov predictors using 4-state confidence automata.

Table 9 compares the same predictors (HMM, Markov, 5 Markovs) without using the confidence automata. In this case, however, the best number of hidden states was 1. We obtained (for all benchmarks) lower prediction accuracies when we used 5 simple Markov predictors, even related to the simple Markov predictor.

Benchmark	HMM of order 1 (N=1)	Markov of order 1	5 Markov predictors of order 1
Employee 1	91.89	81.98	63.46
Employee 2	79.77	74.15	59.77
Employee 3	70.18	64.9	50.57
Boss	79.49	73.64	59.57
AM	80.33	73.66	58.34

Table 9. Comparing a HMM of order 1 with Markov of order 1 and a predictor consisting in 5 Markov predictors without using confidence automata.

In order to decrease prediction latency, we pre-trained HMMs on the summer benchmarks [6] and after that we tested them on the longer fall benchmarks [6]. Table 10 presents comparatively the obtained results when we used HMM with 5 hidden states and 4-state confidence automata:

Benchmark	HMM of order 1 (N=5)			Pre-trained HMM of order 1 (N=5)		
	Prediction accuracy [%]	Total time [ms]	Prediction Time of the last room [ms]	Prediction accuracy [%]	Total time [ms]	Prediction Time of the last room [ms]
Employee 1	92.68	128074	2053	92.68	3455	40
Employee 2	82.84	620572	4787	84.15	17195	110
Employee 3	76.55	589537	4466	73.37	26578	90
Boss	87.17	489243	4256	84.72	13720	100
AM	84.81	456856.5	3890.5	83.73	15237	85

Table 10. Comparing simple HMMs with pre-trained HMMs, both with 4-state confidence automata, in terms of prediction latency.

As it can be observed, with pre-trained HMMs we obtained better prediction latencies (about 85 ms) but for Employee 3 and Boss we obtained lower prediction accuracies. The same prediction accuracies were obtained using the untrained HMM with only 1 hidden state (see Table 4). Thus, in our opinion, through pre-training, the HMM with 5 hidden states, uses only 1 of the hidden states. We repeated the same simulations without using the confidence automata. Table 11 presents the results for HMM with 5 hidden states and Table 12 presents the results for HMM with 1 hidden state:

Benchmark	**HMM of order 1 (N=5)**			**Pre-trained HMM of order 1 (N=5)**		
	Prediction accuracy [%]	**Total time [ms]**	**Prediction Time of the last room [ms]**	**Prediction accuracy [%]**	**Total time [ms]**	**Prediction Time of the last room [ms]**
Employee 1	91.89	118701	2273	91.89	3175	50
Employee 2	75.65	652768	5158	79.77	16393	120
Employee 3	65.28	620322	4727	70.18	62390	90
Boss	72.8	519497	4517	79.49	13088	111
AM	76.4	477822	4168.75	80.33	23761.5	92.75

Table 11. Comparing simple HMMs with pre-trained HMMs, both without confidence automata, in terms of prediction latency (N=5).

Benchmark	**HMM of order 1 (N=1)**			**Pre-trained HMM of order 1 (N=1)**		
	Prediction accuracy [%]	Total time [ms]	**Prediction Time of the last room [ms]**	**Prediction accuracy [%]**	Total time [ms]	**Prediction Time of the last room [ms]**
Employee 1	91.89	471	20	91.89	441	10
Employee 2	79.77	2393	20	79.77	2003	10
Employee 3	70.18	2333	20	70.18	2594	10
Boss	79.49	1963	20	79.49	2234	10
AM	80.33	1790	20	80.33	1818	10

Table 12. Comparing simple HMMs with pre-trained HMMs, both without confidence automata, in terms of prediction latency (N=1).

As it can be seen in Table 11, for a HMM with 5 hidden states through pre-training there are obtained better prediction latencies and better prediction accuracies. It is interesting that we obtained the same prediction accuracies using the untrained HMM with 1 hidden state (see Table 7, N=1). And it is more interesting that for HMM with 1 hidden state, after pre-training, the prediction accuracy is the same again. These results encourage our previous conclusion, that the best number of hidden states is 1, when we don't use confidence automata. Through pre-training, the HMM with 5 hidden states and without confidence automata, uses only 1 of the hidden states.

The next parameter we varied was the order of HMM. We studied HMMs of different orders and the same number of hidden states (N=2). As it can be observed in Table 13, with a first order HMM were obtained the best prediction accuracies:

BENCHMARK	**With 4-state confidence**			**Without confidence**		
	R=1	**R=3**	**R=5**	**R=1**	**R=3**	**R=5**
EMPLOYEE 1	92.68	83.11	80.76	91.89	81.98	84.68
EMPLOYEE 2	84.06	64.06	62.30	77.9	61.04	58.80
EMPLOYEE 3	73.82	63.63	53.33	65.66	53.58	50.56
BOSS	84.72	76.15	73.27	79.07	61.08	60.25
AM	83.82	71.73	67.41	78.63	64.42	63.57

Table 13. Studying HMMs of different orders.

5. Conclusions and Further Work

This paper analyzed machine learning techniques based on HMMs, used in a ubiquitous computing application. Our goal was to predict accurately the movements of persons within an office building. Two predictor types were analyzed: HMMs with confidence automata and respectively without confidence automata. The experimental results show that HMMs outperform other implemented prediction techniques such as Neural Network and respectively Markov predictors. The evaluations show that the simplest configuration of HMM (N=1 and R=1, equivalent with a simple Markov model of order 0) is the most accurate for this specific application. We continued our study implementing a statically pre-trained HMM and we obtained lower prediction latencies.

Predicting from all rooms except own room and using a HMM with 4-state confidence automata, we obtained an average prediction accuracy of 84.81%, but the prediction accuracy measured on some local predictors grew up to over than 92%. As a further work we intend to use our NN and HMM developed predictors in other context prediction in ubiquitous computing applications (e.g., next cell movements in GSM communications based on Nokia's benchmarks).

Acknowledgments

This work was supported in part by the Romanian National Council of Academic Research (CNCSIS) through the grant CNCSIS 71 / 2004 - 2006. Also our gratitude to Professor Theo Ungerer from the University of Augsburg, Germany, for providing us the used benchmarks for persons' movements and for supporting the second author during a two months research stage at Augsburg in 2003.

REFERENCES

1. BIRNEY E., **Hidden Markov Models in Biological Sequence Analysis**, IBM Journal of Research and Development, Volume 45, Numbers 3/4, 2001.
2. GALATA A., JOHNSON N., HOGG D., **Learning Behaviour Models of Human Activities**, Proceedings of British Machine Vision Conference, pages 12-22, Nottingham, UK, September 1999.
3. LIU N., LOVELL B. C., **Gesture Classification Using Hidden Markov Models and Viterbi Path Counting**, Proceedings of the Seventh International Conference on Digital Image Computing: Techniques and Applications, pages 273-282, Sydney, Australia, December 2003.
4. PETZOLD J., BAGCI F., TRUMLER W., UNGERER T., **Context Prediction Based on Branch Prediction Methods**, Technical Report, University of Augsburg, Germany, 2003.
5. PETZOLD J., BAGCI F., TRUMLER W., UNGERER T., VINTAN L., **Global State Context Prediction Techniques Applied to a Smart Office Building,** Communication Networks and Distributed Systems Modeling and Simulation Conference, San Diego, California, SUA, January 18-21, 2004
6. PETZOLD J., **Augsburg Indoor Location Tracking Benchmarks**, Technical Report 2004-9, Institute of Computer Science, University of Augsburg, Germany, 2004, http://www.informatik.uni-augsburg.de/skripts/techreports/.
7. RABINER L. R., **A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,** Proceedings of the IEEE, Vol 77, No. 2, February 1989.
8. STAMP M., **A Revealing Introduction to Hidden Markov Models**, January 2004, http://www.cs.sjsu.edu/faculty/stamp/RUA/HMM.pdf.
9. VINTAN L., GELLERT A., PETZOLD J., UNGERER T., **Person Movement Prediction Using Neural Networks,** Proceedings of the KI2004 International Workshop on Modeling and Retrieval of Context (MRC 2004), Vol-114, ISSN 1613-0073, Ulm, Germany, September 2004.
10. YOON B., VAIDYNATHAN P. P., **RNA Secondary Structure Prediction Using Context-Sensitive Hidden Markov Models**, Proceedings of International Workshop on Biomedical Circuits and Systems, Singapore, December 2004.