Rejection of Irrelevant Human Actions in Real-time Hidden Markov Model based Recognition Systems for Wearable Computers

Jerry Mannil, Mohammad-Mahdi Bidmeshki and Roozbeh Jafari
Embedded Systems and Signal Processing Lab
The University of Texas at Dallas, Richardson, TX 75080-3021
{jerrymannil, bidmeshki, rjafari}@utdallas.edu

ABSTRACT

Hidden Markov Model (HMM) is a well established technique for detecting patterns in a stream of observations. It performs well when the observation sequence does not contain unseen patterns that were not part of the training set. In an unconstrained environment, the observation sequence might contain new patterns that the HMM model is not familiar with. In such cases, HMM will match the test pattern to some trained pattern, which is most similar to the test pattern. This increases the false positives in the system. In this paper, we are describing a threshold based technique to detect such irrelevant patterns in a continuous stream of observations, and classify them as unwanted or bad patterns. The novelty of our approach is that it allows early detection of unwanted patterns. Test patterns are validated on a fixed length substring of observation sequence, rather than on the whole observation sequence corresponding to the test pattern. The substrings are validated based on its similarity with a valid pattern using a threshold value. This reduces the latency of detection of unwanted movement, and makes the detection process independent of duration of the various pattern classes. We evaluated this technique in the context of body sensor networks based human action recognition, and have achieved about 93 percent accuracy in detecting unwanted actions, while maintaining a 94 percent accuracy of recognizing valid actions.

Categories and Subject Descriptors

I.5.2 [**PATTERN RECOGNITION**]: Design Methodology—*Classifier design and evaluation*

General Terms

Design, Algorithms, Experimentation

Keywords

Wearable Computing, Body Sensor Networks, Hidden Markov Model, Action Recognition

Wireless Health '11, October 10-13, 2011, San Diego, USA

1. INTRODUCTION

Body Sensor Networks (BSNs) are becoming increasingly popular due to their applications in health care [14, 19, 24]. BSNs consist of one or more wearable sensor nodes connected wirelessly to a personal device (PDA, Smart Phone or PC) via a base station or actuator node. A variety of sensors is available which can measure different characteristics of the human body. For example physiological sensors measure the heart rate, blood pressure, respiration rate or brain activity, while inertial sensors measure the acceleration and rotation of human body parts. Wearable sensors systems have a significant potential in providing quality health care at a reasonable cost. It helps in continuous monitoring of patients in hospitals in a non-invasive way. Together with mHealth or tele-medicine, they facilitates rehabilitation and assisted living by extended monitoring of the patients in home or outdoor environments [18]. Apart from healthcare, BSN have applications in sports training [9, 7], assisted surgery [1] and brain-machine interfaces [20].

Real-time monitoring systems need to analyze continuous stream of sensor data and extract meaningful information out of it. This often involves finding the spatial and temporal patterns from the sensor data that can characterize the interested phenomena. The first phase involves training the system to learn these patterns from a finite set of sensor data. For example in a patient monitoring system with inertial sensors attached to the patient's body, the system needs to learn the patterns of accelerometer and/or gyroscope readings corresponding to various movements performed by the patient [11]. The second phase (classification) involves locating these patterns of interest in a new stream of observations. Classification often involves finding the stored template pattern that matches most closely with the test pattern using some similarity score.

In an unconstrained environment, it is quite natural that the input signal can consist of unseen observations that were not accounted for in the training phase. In case of an action recognition system, this can happen due to some unexpected movements performed by the subject. It is incongruous to put a constraint that the subject will always performs movements that were already seen in the training phase. Statistical based classification systems classify an unseen pattern with one of the stored pattern templates, even if the similarity is minimal. This can increase the false positives in the recognition system. High false positive rates can make the recognition system unreliable and can add to the cost. For example cost due to servicing of false alarms. Hence, it is

extremely important for the recognition systems to detect and reject these unwanted observations.

Rejection of unwanted patterns is usually addressed by placing an additional threshold constraint to the similarity score. So it is not only sufficient for a test pattern to be similar to a template pattern, but the similarity score should be above the threshold value. All current approaches calculate this similarity score on the complete observation sequence corresponding to a test pattern. This can induce a delay in the recognition process which is not acceptable for real-time recognition systems. For example a lie-to-sit movement in an elderly subject can take about 10-15 seconds. So the observation sequence corresponding to this movement can only be validated after 10-15 seconds from the start of the movement, resulting in a recognition delay of 10-15 seconds. Similarly a sit-to-stand movement takes about 5-8 seconds. So the process of getting out of bed for an elderly subject takes about 15-22 seconds. One of the important causes for falls among patients at hospitals is the fact that patients often try to get out of their beds on their own. So for patients with fall risk, delay in action validation means delay in providing assistance. Also different patterns have different observation sequence lengths. For example sit-to-stand movement can take about 5-8 seconds, while sit-to-lie movement takes about 10-15 seconds. So the detection latency is also dependent on the duration of the patterns. Therefore, early validation of actions, before seeing the full observations can reduce the latency of the recognition system and have potential application in fall prevention systems. Clinicians can track the activities of in-patients at hospitals in real-time and be alerted of any abnormal activities of the patients that can lead to potential accidents such as falls.

In this paper, we describe a method to achieve early detection of unwanted movements, in a real-time Hidden Markov Model [28] based human action recognition system using wearable inertial sensors. Our approach is novel in the following aspects: (1) Validation of actions based on fixed-length substring of the observation sequence corresponding to the test actions (2) Fixed detection latency, independent of the pattern duration (3) Use of similarity score in validating human actions in a continuous stream of data. The substrings are validated based on its similarity with the valid (known) patterns using a threshold value. The similarity score for each valid pattern is calculated over a moving window on the pattern, so that a score can be determined for the fixed length substring of the valid pattern. The similarity score is based on the forward procedure [28] of the HMM and is determined during the training phase of the HMM. Viterbi decoding is used to segment and classify a test pattern in the continuous observation sequence. The forward procedure is then used to validate the test pattern. We evaluated this technique in our recognition system, and have achieved about 93 percent accuracy in detecting unwanted human action, while maintaining a 94 percent accuracy of recognizing valid actions.

2. RELATED WORK

Human activity or action recognition is one of the important applications of BSNs. Body worn inertial sensors can be used to infer the actions performed by the subjects. Considerable amount of work has been done in the past towards using statistical pattern analysis to perform activity detection on body sensor data. Literature cites the use of Decision tree [25, 3], Artificial Neural Network (ANN) [3, 12], k-Nearest Neighbor (k-NN) [27, 6] and hidden Markov Model (HMM) based system to achieve action recognition with body sensor data [11, 15, 10, 41]. Power optimized techniques for analyzing body sensor data has been studied by Authors in [6, 8, 37].

Hidden Markov Model (HMM) [28] is the most widely used mechanism for activity recognition based on body sensor data. HMM is a well established technique for pattern recognition and has also been successfully applied in other domains such as speech recognition [33, 40], signature verification [38, 42] and gesture recognition [2, 21]. Baum-Welch algorithm is usually used to train the HMM based systems for learning relevant patterns. Viterbi algorithm is then used for pattern spotting in a test stream of data [28]. Standard Viterbi algorithm needs to see the full observation sequence before emitting the most probable state sequence. This is not suitable for real-time decoding as the observation sequence is generated in a continuous manner, and a new observation is generated for each instance of sample time. The system may run for hours or even days generating huge amount of observation data. It is neither practical to store this large amount of data, nor it is feasible to wait that long to start the recognition process. Literature describes the use of a window based Viterbi procedure [2, 26] to overcome this problem. The Viterbi decoding is performed based on a fixed number of past observations, rather than the full observations. For this we need to maintain a history buffer, to keep track of the past observations, and the state sequences corresponding to these observations.

Rejection of unrelated patterns is essential for the success of any practical recognition system. This problem has been well studied in the domain of speech and handwriting recognition. Several techniques have been proposed to address this issue. Authors in [41] have given a good summary of various techniques used in prior literature to detect the occurrence of unwanted patterns. They categorized the rejection techniques into three categories based on how the unwanted patterns are modeled.

The first approach is to build a set of models to represent unwanted patterns, called as filler or garbage models. The garbage models are trained using a finite number of unwanted patterns. This approach is widely used in speech recognition systems and is used to model acoustic non-keyword segments using Hidden Markov Model (HMM) [33, 40]. Authors in [17] used this approach to detect non-keyword segments in Dynamic Time Warping (DTW) based speech recognition system. They created "filler templates" to represent non-keyword speech segments. This approach is not particularly suitable for gesture or action recognition. There can be an arbitrarily large number of unwanted gesture or action patterns, and it is difficult to represent them with just a few filler models. Filler model based detection of unwanted human actions have been achieved with limited success in [2]. They used a set of commonly occurring non-gesture patterns to build the garbage model.

The second approach is to extrapolate the model of unwanted patterns based on the models of valid patterns [31], and is widely used in speaker verification systems [31, 16]. The likelihood of the unwanted pattern was calculated as a function of the likelihoods of valid patterns. A pattern is accepted as a valid pattern or not, based on the outcome

of the likelihood ratio test [31] using a pre-defined threshold value. This method assumes that the likelihood of valid patterns is higher than that of the estimated unwanted pattern, and uses this idea to reject unwanted patterns. There are various flavors to this approach based on the function used to estimate the likelihood of valid patterns. Average [34] and maximum [16] functions are commonly used for this purpose. This approach has also been used to reject non-gesture patterns and non-meaningful activities, based on the multi-variate distribution of the likelihood of valid patterns [41]. This approach also has the added benefit of not having to explicitly model the unwanted patterns. However, it is possible that an unwanted pattern can have a likelihood similar to that of a valid pattern, though the probability of such occurrence is much less.

The third approach is to train a set of models to represent unwanted patterns using the meaningful patterns itself. They are called background models and are extensively used in speaker [36, 31] and signature verification systems [22, 35]. In [31], background models were trained using Gaussian mixture models from the speech samples collected from various speakers. Authors in [22] built a Gaussian mixture model based background model for signature verification. Background model based on Hidden Markov Model was used in [36] for speaker verification and [35] for signature verification. Likelihood ratio test was usually used for pattern validation with this approach. Authors in [21] describe a method called Threshold Hidden Markov Model (THMM) to reject unwanted gestures, where the background model was also based on HMM. The threshold HMM model was built by combining the individual HMM models of each meaningful gesture. The THMM approach determines the threshold for validation of a pattern dynamically in contrast to other approaches.

The fourth approach calculates an individual rejection threshold for each valid pattern by using only the likelihoods of training samples of the respective patterns. The threshold for a valid pattern defines the minimum likelihood that the training samples of the valid pattern must exhibit in its own trained model. Any test pattern matched to a meaningful pattern must have its likelihood greater than that of the threshold. Otherwise the pattern is rejected. This method has been successfully applied in Hidden Markov Model [38, 42, 43] and Gaussian Mixture Model [38, 32, 23] based signature verification systems with good results. This approach has also been used in Dynamic Time Warping based [23] signature verification and speech recognition [29] systems.

There are other approaches that can complement the above mentioned approaches, in the rejection of unwanted patterns. One such approach is known as "duration test" [39, 38], which put a minimum cap on the duration or the length of the test patterns. Patterns with short length tend to have large likelihoods, even though they may not be of interest (unwanted) [38]. Hence, any pattern with duration less than the minimum threshold is rejected. Another approach known as "energy level test" [39] validated the speech utterance based on the normalized energy level of the speech signal.

All of the above approaches perform the pattern validation after completely observing the relevant patterns. In our approach, we try to detect unwanted patterns based on partial observation of the patterns.

3. SYSTEM OVERVIEW

3.1 Data Collection

In this section, we describe the body sensor network used to collect the human inertial data, and the human actions we are interested in.

3.1.1 Actions

Sensor data was collected for the actions shown in Table 1. We were interested in detecting actions from 1 to 13. These actions were chosen as we are developing a system to monitor in-patients in real-time at hospitals. Our action set consist of two categories: postures and movements. A posture represents a still posture of human body. Actions 9-13 represent postures. On the other hand, movements are dynamic postures which involve actual movement of human body parts. Actions 1-8 represent movements. A movement is assumed to start and end with some static postures. The HMM used for action recognition will be trained using sensor data corresponding to these actions. Actions 21-27 represent a set of unwanted or unknown movements. These movements were not used for training the system. Unwanted actions can be a static posture, movement or a combination of both. These movements must be detected as unwanted actions by the system.

3.1.2 Sensor hardware

Figure 2 shows one of the sensor nodes (motes) used to collect data for our experiments. The sensor node consists of commercially available moteiv TelosB as the microcontroller with a custom-designed sensor board and is powered by a rechargeable 3.7V, 2.3Wh Li-ion battery. The processor is a 16 bit, 8 MHz TI MSP430. The sensor board includes a tri-axial accelerometer and a bi-axial gyroscope.

3.1.3 Mote Placement

We used two sensor nodes in our recognition system. The motes were placed on the body as shown in Figure 1. One mote was placed on waist and the other on right thigh. These placement of motes were sufficient to differentiate between the movements we were interested in.

3.2 Signal Processing

In this section we discuss the various steps involved in the action recognition process. It involves a number of signal processing tasks that execute on the sensor nodes and the basestation as shown in Figure 3. The data is processed on a moving window centered on the current sample. The window moves forward one sample at a time.

3.2.1 Sensor Data

Each mote measures five sensor values at each sample time. The accelerometer senses three axes of acceleration. The gyroscope senses angular velocity in two axes along the plane of the sensor board. Data is collected from each mote at a sampling rate of 20 Hz. However, we only used the accelerometer sensor data to build our recognition system.

3.2.2 Feature Extraction

For each sample time, a feature vector is generated from the raw sensor data. A feature can better represent the measurable properties of the sensor data. The following features are extracted from a five sample window for each sensor:

Table 1: Movement Set.

No	Action	Start Posture	End Posture
1	Lie-to-LieLeft (L2LLt)	Lie	LieLeft
2	LieLeft-to-Lie (LLt2L)	Lieleft	Lie
3	Lie-to-LieRight (L2LRt)	Lie	LieRight
4	LieRight-to-Lie (LRt2L)	LieRight	Lie
5	Lie-to-Sit (L2St)	Lie	Sit
6	Sit-to-Lie (St2L)	Sit	Lie
7	Sit-to-Stand (St2Snd)	Sit	Stand
8	Stand-to-Sit (Snd2St)	Stand	Sit
9	Lie (Lie)		
10	LieLeft (LieLt)		
11	LieRight (LieRt)		
12	Sit (Sit)		
13	Stand (Snd)		
21	Turn clockwise 90 (Trn90)		
22	Turn anti-clockwise 90 (TrnA90)		
23a	Move Forward 1 step(MvFwd)		
23b	Move Backward 1 step(MvBwd)		
24	Move Sideways (Mvsdwy)		
25	Bend Forward (BndFwd)		
26	Sit On Floor (SitFlr)		
27	Lift right leg (LftRtLg)		

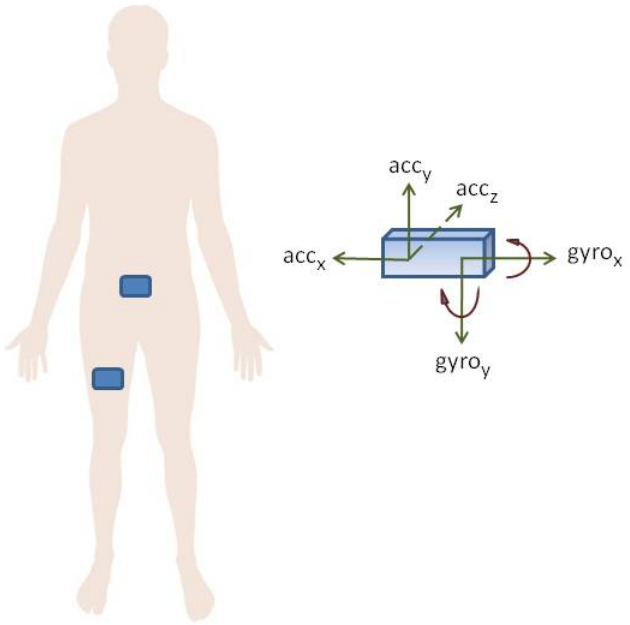

Figure 1: Mote Placements

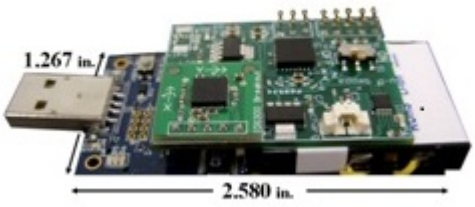

Figure 2: TelsoB Mote

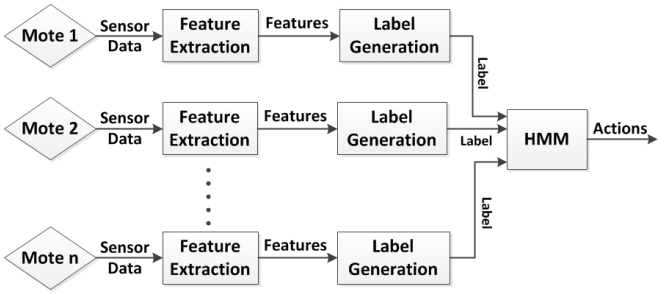

Figure 3: Signal Processing

mean, standard deviation, second derivative. The window moves one sample at a time.

3.2.3 *Label Generation*

Labels are characters from a small alphabet which can concisely represent the entire feature set. Statistical pattern recognition algorithms like the Hidden Markov Model (HMM), work with scalar (dimension of one) observations. So it is necessary to generate scalar observations or labels out of multi-dimensional feature vectors such that the labels characterize the entire feature space.

Transmitting labels instead of the raw data can significantly improve the battery performance of the sensor nodes. Raw data generated from various sensors at each sample time carry a lot of information. In our case each sensor reading is 2 bytes. So the raw sensor data at each sample time is 3*2 = 6 bytes. At a sampling frequency for 20 Hz, this translates to 120 bytes of data generated from each sensor node every second. In a sensor node, the wireless communication cost dominates over the local computational cost with respect to the battery consumption [13, 30]. So instead of transmitting the entire feature vector, each mote sends the label corresponding to that feature vector to the basestation. We used Gaussian Mixture Models (GMM) to generate labels based on clusters generated from the feature space [5]. We used an alphabet with twenty characters to generate labels for each motes. Each label takes half a byte. So the data transmission rate effectively reduces to 0.5*20 = 10 bytes/second.

3.2.4 *Hidden Markov Model*

The label sequence corresponding to each action is analyzed to extract relevant patterns. HMM was used to learn the label patterns for different movements, as described in [11]. Each action was modeled by a separate left-to-right HMM with one or five states. Static postures were modeled by a single state HMM, while the dynamic postures were modeled with five HMM states. The continuous action recognizer was formed by joining all these individual HMMs to form a recognition network as shown in Figure 4. The static actions form the source and sink nodes in the network, with the dynamic actions as internal nodes. Baum-Welch [28] algorithm was used to train each individual HMMs and form the HMM network.

3.2.5 *Classification*

We used the Viterbi algorithm to segment and classify the label sequence into the corresponding human actions.

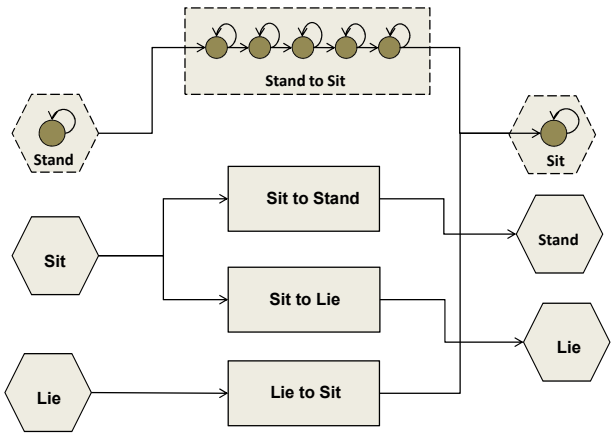

Figure 4: HMM recognition network

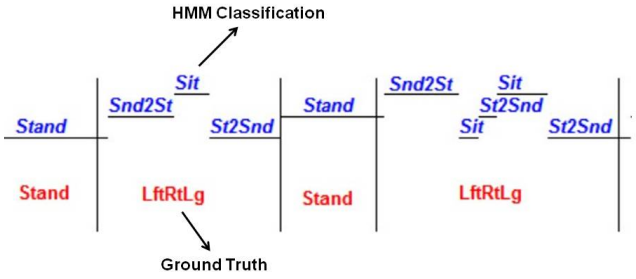

Figure 5: Viterbi output of unknown action 27

4. REJECTION MECHANISM

4.1 Problem Description

The above described system performs well in identifying actions in a noise free environment. By noise free environment, we mean that the human subject is expected to perform only the movements that are part of the training set, i.e., Actions 1-13. If a subject performs movements that are not part of the training set, the system identifies it as a combination of one or more movements in the training set. This increases false positives in the system. Figure 5 depicts the classification of movement 27. The horizontal lines and blue labels (in italic) represent the Viterbi output; whereas the vertical line and the red labels depict the true actions performed by the subject. The system exhibits such a behavior due to the forced classification nature of Viterbi decoding. Viterbi decoding performs classification based on the Maximum Likelihood (ML) criterion. Therefore the test pattern is always classified as one of the known patterns, even if the test pattern does not belong to any of the known pattern classes [41].

4.2 Threshold estimation

We extend the approach in [38] to reject unwanted actions. We first use the training samples to train the HMM model for each action using the Baum-Welch procedure as described in [11]. A training sample (trial) is an observation sequence that characterizes a particular action. We then estimate the likelihoods of training observation sequence on that HMM model and use it to derive the threshold value for each HMM. The forward procedure of HMM [28] allows us to calculate this likelihood.

Given an observation $O = \{O_1 O_2 ... O_T\}$, its likelihood with respect to an HMM λ is denotes by $P(O|\lambda)$. To calculate $P(O|\lambda)$, we need to define the forward variable $\alpha_t(i)$ as

$$\alpha_t(i) = P[O_1 O_1 ... O_t, q_i = S_i|\lambda] \quad 1 \leq t \leq T, \quad 1 \leq j \leq N$$

This gives the probability of the partial observation sequence $O_1, O_2, ...O_t$ and state S_i at time t, given the HMM λ with N hidden states. This can be efficiently computed as follows:

1. Initialization

$$\alpha_1(i) = \pi(i) b_i(O_1) \quad 1 \leq i \leq N \tag{1}$$

2. Recursion

$$\alpha_t(j) = \left[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(O_t) \quad 2 \leq t \leq T, \ 1 \leq i \leq N \tag{2}$$

3. Termination

$$P[O|\lambda] = \sum_{i=1}^{N} \alpha_T(i) \tag{3}$$

Observation sequences for the same action can have different lengths. Observation sequences with shorter length tend to have large likelihoods [39, 38]. In order to make them comparable to each other, we need to normalize the observation probabilities calculated in (3). Literature suggests the use of duration normalized likelihood to overcome the noise introduced due to variation in sequence length [4, 33]. Duration normalized likelihood $P_n(O|\lambda)$ was computed as

$$P_n[O|\lambda] = log\left(P[O|\lambda]\right)/T \tag{4}$$

Figure 6 depicts the normalized log probability distribution for actions 1, 2, 9 and 10. Each graph in the figure represents the distribution with respect to the HMM model of a particular action. The Y-axis represents the normalized observation probabilities, whereas the X-axis depicts the training sample number. The red line with asterisk markers show the probabilities of training samples for a particular action over its own HMM model. Each asterisk represents a trial or training sample. The blue lines with dot markers show the corresponding probabilities over training samples of other actions. The number at the end of each line depicts the action that this line represents. Clearly, the likelihoods of each action observation sequences on its own HMM model is comparatively high with respect to the training samples of other actions. This helps us in determining threshold likelihood for each valid action using the similarity score assigned to them. Any action classified using the Viterbi decoding is validated using the threshold. If the likelihood of the recognized action is less than the threshold, we detect the action as unwanted. The similarity score represents the minimum likelihood over all the training samples for an action. We define the similarity score $S(h)$ for an action h as:

$$S(h) = min\left\{P_n(O^k|\lambda_h)\right\} \quad \forall\ O^k \in TS_h \tag{5}$$

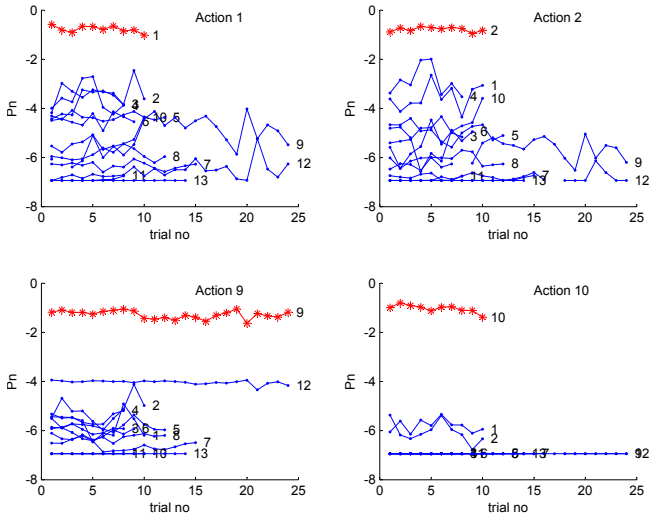

Figure 6: Probabilities over whole training samples

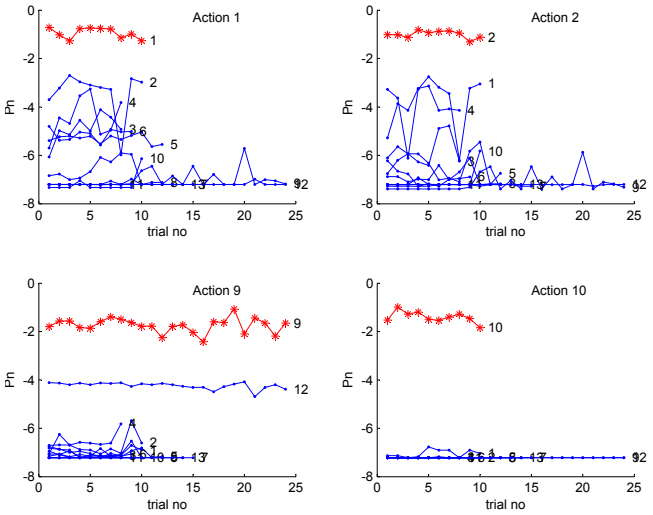

Figure 7: Probabilities over one second window, minimum

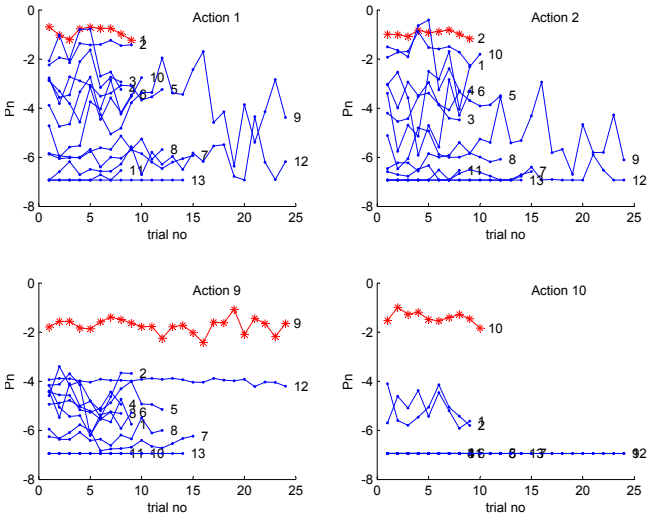

Figure 8: Probabilities over one second window, maximum

where TS_h is the training set for action h.

But this similarity score is calculated over the entire observation sequence corresponding to an action. This is not appropriate for a real-time recognition system as it can incur latency in detection of unwanted movements.

Hence, we estimated the observation probabilities over a moving window to see if the probabilities of fixed length substrings within the observation sequence of an action are discriminative enough to validate an action. We calculated the normalized probabilities for each training sample over a window of one second, with the window moving one sample at a time. Figure 7 depicts the minimum probability distributions. Each dot or asterisk represents the minimum of all the probabilities of the substrings generated over the moving window for a training sample. Clearly, there are substrings for trials of each actions whose likelihoods with respect to its own HMM model is comparatively high with respect to the substrings for trials of other actions.

Figure 8 depicts the minimum probability distributions over a moving window of one second for an action with respect to the maximum probability distributions for other actions. As before, red line with asterisk markers represent the minimum probability of the training samples of an action over its own HMM. However, each blue line represent the maximum of all the probabilities of the substrings generated by the moving window for training samples of other actions. Figure 8 shows that there are instances where the maximum probabilities of fixed length substrings of other actions with respect to the HMM of a particular action, can be close or exceed the minimum probabilities of the actions corresponding to that HMM. This can happen since sub-patterns among different actions can be similar in some instances of the moving window.

1. The likelihoods of the substrings of all other actions with respect to HMM of a particular action is well below the likelihood value for most instances in the moving window. Thus, it does not hinder us from detecting the occurrence of an unwanted action.

2. Each human action occurs for a minimum amount of time. In our case, it is about one second. Viterbi decoding of unwanted actions often results in fast transition between HMMs as shown in figure 5. This result in action sequences with very short durations. Patterns with shorter length tend to have large likelihoods, even though they may not be of interest [38]. Any Viterbi decoding with smaller action duration (less than one second) is potentially due to an unwanted action, and can be rejected.

3. The static postures (for example, actions 9, 10) have very discriminating likelihoods over the substrings. This is evident from Figures 7 and 8. The probabilities of all the substrings of the static postures are well above the probabilities of other actions (static or dynamic). This is the case because static postures have a simple pattern (more or less a straight line) and its structure is not affected by the sequence length. Every dynamic action needs to make a transition to a static posture

in a finite amount of time. Hence, once an unwanted dynamic action is detected, we ignore any higher likelihood observations until a valid static posture is detected.

Finally, the threshold likelihood of each action is set as a constant factor C (greater than one) times the similarity score (5) of the corresponding action. This factor was determined empirically such that the recognition accuracies of valid actions and the rejection accuracies of unwanted actions are within reasonable limits.

$$Threshold[h] = C * S(h) \tag{6}$$

4.3 Algorithm

In this section, we describe how we adapted the idea mentioned in the previous section to perform rejection of unwanted movements on a continuous stream of data. First, we run the Viterbi algorithm on the incoming observations for each sample. We use a history of past three seconds of data, to detect the action performed. We validate the currently detected action in the following manner:

1. If the length of the action is less than one second and the current action detected is different the previous one, we assume that the subject has performed an unwanted action and classify the action as unwanted.

2. If the length of action is greater than one second, we calculate the normalized probability over one second of data, or the length of the action, whichever is smaller.

If the probability is greater than the threshold, we infer that the action is valid, and output the currently detected action. Since the probability is calculated over one second of data, we keep the current decision for the next one second of Viterbi decoding. After one second has elapsed, we again check for the validity of the currently detected action.

If the probability is less than the threshold, we detect the action as unwanted. We then reject all actions, until we see the next valid static posture. The threshold value is set as a constant factor times the similarity score.

4.4 Computational complexity

The rejection mechanism performs additional computations to determine the likelihood of the label sequence based on the forward procedure of the HMM. The run-time order of the rejection mechanism is given by

$$O(N \cdot T) \tag{7}$$

where N is the number of states in the action HMM and T is the length of the moving window used for the likelihood calculation. Note that the computational complexity is lesser than that of the standard forward procedure since we used left-right HMM to represent actions. Since the label sequence used to calculate the likelihood is taken from the Viterbi decoding stage, no additional latency is induced by the rejection mechanism.

5. EXPERIMENTAL RESULTS

We trained our system with actions performed by four healthy male subjects. These subjects were asked to perform actions 1 to 13 of Table 1 in ten consecutive trials. Video of the trials has been recorded and was used to manually annotate the start and the end of each action in the continuous stream of data, to generate the training samples for each action. Sixty percent of movement data were used for training and the rest were used for testing the performance of our recognition system in a noise-free environment. The experiments were conducted in a controlled environment at our lab.

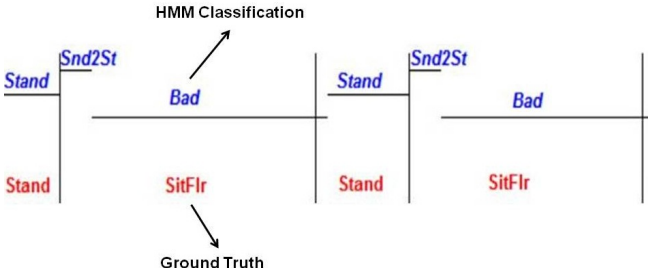

Figure 9: Viterbi output of unknown action 26 with rejection

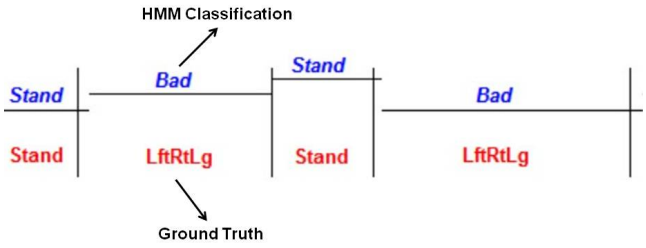

Figure 10: Viterbi output of unknown action 27 with rejection

For validation of action rejection mechanism, we collected ten trials of unwanted actions 21-27 of Table 1 from the above four subjects. Again, the test data containing the unwanted movements were manually annotated to mark the start and end of known and unwanted movements, using the video data. This data along with the training data were used to measure the performance of the recognition system with the rejection mechanism. Figure 9 and 10 show the Viterbi output with rejection for action sequence containing unwanted movements 26 and 27 respectively. The label *Bad* depicts the unwanted actions.

We used the Precision (P) and Recall (R) metric to measure the performance of the system in the presence of unknown movements. These metrics are computed based on true positives (tp), false positives (fp) and false negatives (fn) and are given by

$$P = \frac{tp}{tp + fp} \tag{8}$$

$$R = \frac{tp}{tp + fn} \tag{9}$$

Precision tells how well the system performs in the presence of false positives (due to unwanted actions), and is a good measure to evaluate the action rejection performance. Recall on the other hand tells how well the system recognizes known actions, and is a good measure to evaluate the

Table 2: Precision

Action	Without rejection (%)	With rejection (%)
Lie-to-LieLeft (L2LLt)	83	87
LieLeft-to-Lie (LLt2L)	95	95
Lie-to-LieRight (L2LRt)	91	90
LieRight-to-Lie (LRt2L)	94	94
Lie-to-Sit (L2St)	72	94
Sit-to-Lie (St2L)	75	100
Sit-to-Stand (St2Snd)	20	91
Stand-to-Sit (Snd2St)	28	69
Lie (Lie)	100	100
LieLeft (LieLt)	100	100
LieRight (LieRt)	100	100
Sit (Sit)	100	100
Stand (Snd)	100	100
Known	73	96
Unknown (Bad)	NA	88

Table 3: Recall

Action	Without rejection (%)	With rejection (%)
Lie-to-LieLeft (L2LLt)	100	100
LieLeft-to-Lie (LLt2L)	100	90
Lie-to-LieRight (L2LRt)	100	95
LieRight-to-Lie (LRt2L)	100	100
Lie-to-Sit (L2St)	100	100
Sit-to-Lie (St2L)	100	93
Sit-to-Stand (St2Snd)	100	100
Stand-to-Sit (Snd2St)	100	100
Lie (Lie)	97	97
LieLeft (LieLt)	81	81
LieRight (LieRt)	90	90
Sit (Sit)	96	96
Stand (Snd)	92	92
Known	95	94
Unknown (Bad)	0	93

action recognition performance.

The threshold value used to validate an action was calculated as a constant factor (greater than one) times the similarity score for the action. This factor was empirically chosen so as to maintain a balance between the recognition and detection accuracies. A smaller factor (closer to one) will improve the detection accuracy, but at the cost of lower recognition accuracy. A very high value for the factor (closer to two) will increase the recognition accuracy, but at the cost of lower detection accuracy.

Tables 2 and 3 give the performance metric of the system for detecting various actions described in Table 1. The action *Known* represents the combined results of the valid movements 1-13. The action *Unknown* represent the unwanted movements 21-27. We see that the precision of known movements 1, 5, 6, 7 and 8 were greatly increased by the introduction of rejection mechanism. This is because the false positives due to unwanted movements were identified correctly and classified as unwanted movements. Also we achieved a good accuracy of 93 percent in detecting unwanted movements. However, the recognition accuracy of known movements reduced slightly, due to the misclassification of some valid movements as unwanted.

6. CONCLUSIONS

In this paper, we have presented a mechanism to detect unwanted actions in a continuous real-time HMM based action recognition system. Our approach allows early detection of unwanted actions, inducing no additional latency to the system. We have achieved 93 percent accuracy in detecting unwanted movement while maintaining a recognition accuracy of 94 percent. We verified the real-time rejection mechanism in simulations done in MATLAB. The next step is to incorporate this mechanism in our real-time action recognition system. Also, this method was developed on the inertial data of human activities. We use some properties of the static postures to handle some special cases where the likelihoods of unwanted patterns are higher. But, we expect this technique to work well for a broad set of patterns obtained from different sensors, with or without minor tweaks.

7. ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation, under grants CNS-1138396 and CNS-1012975, the National Institute of Health, under grant R15AG037971, and the funding mechanism provided by the Texas Health Resources and Texas Instruments. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding organizations.

8. REFERENCES

[1] W. T. Ang, P. Khosla, and C. Riviere. Design of all-accelerometer inertial measurement unit for tremor sensing in hand-held microsurgical instrument. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 2, pages 1781 – 1786 vol.2, September 2003.

[2] S. Eickeler, A. Kosmala, and G. Rigoll. Hidden markov model based continuous online gesture recognition. In *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*, volume 2, pages 1206 –1208 vol.2, August 1998.

[3] M. Ermes, J. Parkka, J. Mantyjarvi, and I. Korhonen. Detection of daily activities and sports with wearable sensors in controlled and uncontrolled conditions. *Information Technology in Biomedicine, IEEE Transactions on*, 12(1):20 –26, jan. 2008.

[4] J. Fierrez-Aguilar, J. Ortega-Garcia, and J. Gonzalez-Rodriguez. Target dependent score normalization techniques and their application to signature verification. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 35(3):418 –425, August 2005.

[5] C. Fraley, Adrian, and E. Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. *The Computer Journal*, 41:578–588, 1998.

[6] H. Ghasemzadeh, E. Guenterberg, K. Gilani, and R. Jafari. Action coverage formulation for power optimization in body sensor networks. In *Design*

Automation Conference, 2008. ASPDAC 2008. Asia and South Pacific, pages 446 –451, march 2008.

[7] H. Ghasemzadeh and R. Jafari. Coordination analysis of human movements with body sensor networks: A signal processing model to evaluate baseball swings. *Sensors Journal, IEEE*, 11(3):603 –610, March 2011.

[8] H. Ghasemzadeh and R. Jafari. Ultra low power granular decision making using cross correlation: Optimizing bit resolution for template matching. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2011 17th IEEE*, pages 137 –146, april 2011.

[9] H. Ghasemzadeh, V. Loseu, and R. Jafari. Wearable coach for sport training: A quantitative model to evaluate wrist-rotation in golf. *Ambient Intelligence and Smart Environments, Journal of*, 1:173–184, 2009.

[10] E. Guenterberg, H. Ghasemzadeh, and R. Jafari. A distributed hidden markov model for fine-grained annotation in body sensor networks. In *Proceedings of the 2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*, BSN '09, pages 339–344, Washington, DC, USA, 2009. IEEE Computer Society.

[11] E. Guenterberg, H. Ghasemzadeh, V. Loseu, and R. Jafari. Distributed continuous action recognition using a hidden markov model in body sensor networks. In *Proceedings of the 5th IEEE International Conference on Distributed Computing in Sensor Systems*, DCOSS '09, pages 145–158, Berlin, Heidelberg, 2009. Springer-Verlag.

[12] N. Györbíró, k. Fábián, and G. Hományi. An activity recognition system for mobile phones. *Mobile Networks and Applications*, 14:82–91, 2009. 10.1007/s11036-008-0112-y.

[13] M. Hanson, H. Powell, A. Barth, K. Ringgenberg, B. Calhoun, J. Aylor, and J. Lach. Body area sensor networks: Challenges and opportunities. *Computer*, 42(1):58 –65, jan. 2009.

[14] Y. Hao and R. Foster. Wireless body sensor networks for health-monitoring applications. *Physiological Measurement*, 29(11):R27, 2008.

[15] J. He, H. Li, and J. Tan. Real-time daily activity classification with wireless sensor networks using hidden markov model. In *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, pages 3192 –3195, aug. 2007.

[16] A. Higgins, L. Bahler, and J. Porter. Speaker verification using randomized phrase prompting. *Digital Signal Processing*, 1(2):89 – 106, 1991.

[17] A. Higgins and R. Wohlford. Keyword recognition using template concatenation. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '85.*, volume 10, pages 1233 – 1236, apr 1985.

[18] R. S. H. Istepanian, E. Jovanov, and Y. T. Zhang. Guest editorial introduction to the special section on m-health: Beyond seamless mobility and global wireless health-care connectivity. *Information Technology in Biomedicine, IEEE Transactions on*, 8(1):405–414, December 2004.

[19] B. Latré, B. Braem, I. Moerman, C. Blondia, and P. Demeester. A survey on wireless body area networks. *Wireless Networks*, 17(1):1–18, January 2011.

[20] M. A. Lebedev and M. A. Nicolelis. Brain-machine interfaces: past, present and future. *Trends in Neurosciences*, 29(9):536 – 546, 2006.

[21] H.-K. Lee and J. Kim. An hmm-based threshold model approach for gesture recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(10):961 –973, October 1999.

[22] M. Martinez-Diaz, J. Fierrez, and J. Ortega-Garcia. Universal background models for dynamic signature verification. In *Biometrics: Theory, Applications, and Systems, 2007. BTAS 2007. First IEEE International Conference on*, pages 1 –6, sept. 2007.

[23] O. Miguel-Hurtado, L. Mengibar-Pozo, M. Lorenz, and J. Liu-Jimenez. On-line signature verification by dynamic time warping and gaussian mixture models. In *Security Technology, 2007 41st Annual IEEE International Carnahan Conference on*, pages 23 –29, oct. 2007.

[24] A. Pantelopoulos and N. Bourbakis. A survey on wearable sensor-based systems for health monitoring and prognosis. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 40(1):1 –12, January 2010.

[25] J. Pärkkä and, L. Cluitmans, and M. Ermes. Personalization algorithm for real-time activity recognition using pda, wireless motion bands, and binary decision tree. *Information Technology in Biomedicine, IEEE Transactions on*, 14(5):1211 –1215, sept. 2010.

[26] M. Pilu. Video stabilization as a variational problem and numerical solution with the viterbi method. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–625 – I–630 Vol.1, june-2 july 2004.

[27] S. Pirttikangas, K. Fujinami, and T. Nakajima. Feature selection and activity recognition from wearable sensors. In *Ubiquitous Computing Systems*, volume 4239 of *Lecture Notes in Computer Science*, pages 516–527. 2006.

[28] L. Rabiner and B. Juang. An introduction to hidden markov models. *ASSP Magazine, IEEE*, 3(1):4 – 16, January 1986.

[29] L. Rabiner and C. Schmidt. Application of dynamic time warping to connected digit recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 28(4):377 – 388, aug 1980.

[30] V. Raghunathan, C. Schurgers, S. Park, and M. Srivastava. Energy-aware wireless microsensor networks. *Signal Processing Magazine, IEEE*, 19(2):40 –50, mar 2002.

[31] D. Reynolds and R. Quatieri, T.F. andDunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing: A Review Journal*, 10(1):19–41, 2000.

[32] J. Richiardi and A. Drygajlo. Gaussian mixture models for on-line signature verification. In *Proceedings of the 2003 ACM SIGMM workshop on Biometrics methods and applications*, WBMA '03,

pages 115–122, New York, NY, USA, 2003. ACM.

[33] R. Rose and D. Paul. A hidden markov model based keyword recognition system. In *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, pages 129 –132 vol.1, April 1990.

[34] A. Rosenberg and S. Parthasarathy. Speaker background models for connected digit password speaker verification. In *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, volume 1, pages 81 –84 vol. 1, May 1996.

[35] E. A. Rúa, D. P.-P. n. López, and J. L. A. Castro. Ergodic hmm-ubm system for on-line signature verification. In *Proceedings of the 2009 joint COST 2101 and 2102 international conference on Biometric ID management and multimodal communication*, pages 340–347, Berlin, Heidelberg, 2009. Springer-Verlag.

[36] O. Siohan, C.-H. Lee, A. Surendran, and Q. Li. Background model design for flexible and portable speaker verification systems. In *Acoustics, Speech, and Signal Processing, 1999. ICASSP '99. Proceedings., 1999 IEEE International Conference on*, volume 2, pages 825 –828 vol.2, March 1999.

[37] P. S. Udaya Shankar, N. Raveendranathan, N. R. Gans, and R. Jafari. Towards power optimized kalman filter for gait assessment using wearable sensors. In *Wireless Health 2010*, WH '10, pages 137–144, New York, NY, USA, 2010. ACM.

[38] L. Wan and B. Wan. On-line signature verification with two-stage statistical models. In *Proceedings of the Eighth International Conference on Document Analysis and Recognition*, ICDAR '05, pages 282–286, Washington, DC, USA, 2005. IEEE Computer Society.

[39] J. Wilpon, C. Lee, and L. Rabiner. Application of hidden markov models for recognition of a limited set of words in unconstrained speech. In *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, pages 254 –257 vol.1, may 1989.

[40] J. Wilpon, L. Rabiner, C.-H. Lee, and E. Goldman. Automatic recognition of keywords in unconstrained speech using hidden markov models. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 38(11):1870 –1878, nov 1990.

[41] L. Xiao-Hui and C. Chin-Seng. Rejection of non-meaningful activities for hmm-based activity recognition system. In *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*, pages 189 –196, April 2006.

[42] L. Yang, B. K. Widjaja, and R. Prasad. Application of hidden markov models for signature verification. *Pattern Recognition*, 28(2):161 – 170, 1995.

[43] H. S. Yoon, J. Y. Lee, and H. S. Yang. An on-line signature verification system using hidden markov model in polar space. In *Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02)*, IWFHR '02, pages 329–, Washington, DC, USA, 2002. IEEE Computer Society.

APPENDIX

A. ACTION RECOGNITION PSEUDOCODE

Algorithm 1 Action recognition procedure

```
input: Observation at current sample time
ouput: HMM and state of current actions

Detect_Action(obs)
{
  // hmm: currently detected action
  // state: HMM state
  // len: length of current action
  hmm, state, len = Online_Viterbi(obs);

  if(isStaticposture(hmm))
  {
    //restart validation process
    valid = 0
  }

  // valid_cnt: samples for which current
  //            decision is valid
  if(valid_cnt <= 0)
  {
    valid_cnt = w
    if (isDynamicposture(hmm) and
      (prev_action == Bad))
    {
      hmm = Bad
    }
    // w: length of window (one second)
    else if (len >= w)
    {
      // normalized probability calculation
      pn = log(P(O[1:w], hmm))/w

      // S: Similarity score array
      // C: constant factor
      // threshold = a * s{hmm]
      if (pn < C*S[hmm])
      {
        // unwanted action
        hmm = bad
      }
    }
    else
    {
      // action less than one second, reject
      valid = len
      if (prev_action != hmm)
      {
        hmm = Bad
      }
    }
  }
  else
  {
    // keep the previous decision for
    // "valid_cnt" samples (one second)
    valid = valid - 1
    if(prev_action == Bad)
    {
      hmm = Bad
    }
  }
  prev_action = hmm
  return hmm, state
}
```