Personal Trainer in Your Pocket

Mohammad Amanzadeh*
Department of Informatics

Constantine Roros*
Department of Electrical and Computer Engineering

Sayed Hadi Hashemi*
Department of Computer Science

**University of Illinois at Urbana-Champaign*
{*amanzad2, roros2, hashemi3*}*@illinois.edu*

Abstract

When starting a new sport or practicing a new exercise program, it is extremely important to learn basic movements the right way, otherwise a long learning period or serious injuries are not unexpected. This highlights the importance of having a professional personal trainer. The problem is continuous access to such a person is expensive and not always possible.

This problem pushes the market to provide custom-designed sport gadgets. These gadgets are attached to the human body or accessories, then collect various data from built-in sensors. After that companion software analyze this data to evaluate the performance of the user.

Despite the promising outcome, there are some limitations that slow down the vast usage of these devices. First, these gadgets are still not affordable for everyone, and second their application is usually limited to one particular sport or even one particular movement, and can not be extended to others.

One way to overcome these limitations is to use more available devices for the training purposes.

In this work we propose a method to develop a personal trainer using common sensors of smartphone or health tracker. A Hidden Markov Model (HMM) has been trained for each action and classify the input motions by comparing the likelihood of HMMs trained on each action. We would also use a function of likelihood to assess the similarity of an action to the baseline.

1. Introduction

An important part of learning a new sport or exercise is understanding the right technique to correctly perform the physical activities involved. This ensures that people who engage in these activities do not get injured, get the most out of their work, and perform their best. In order to achieve such abilities, one usually hires a professional personal trainer. While this may be the best option, it is not the most affordable one. Many people have the desire and potential to learn new physical activities but may not have the resources or opportunity to work with a trainer. We hope that developing an application that helps people learn these activities will allow more people to better learn these activities by using a device (such as a smart watch or smart phone) that has an accelerometer. While these devices are not affordable for everyone, they are easier to access than a personal trainer.

The idea behind the application is that the person will wear the device as they are performing the physical activity. The application will indicate to the user what parts of the movement he or she needs to improve. If a user wears these devices while performing an activity, we can track that users movement using the accelerometer inside the device. In order to use this data to help the user train for a physical activity, there are a number of obstacles we must overcome. We must determine when the activity in question actually starts and ends during the accelerometer recording so we know what accelerometer features are specific to the actual activity. Once we figure out what features belong to each activity, we can use those features to classify the recording to figure out what activity the user is doing. In order to classify this data we developed a Hidden Markov Model for each activity by obtaining training data for each type of movement. Ideally the training data for these models is from accelerometer recordings of a user that is skilled at the given activity. Once we develop the model we can score the performance of the user when they complete the motion by comparing that motion to the recordings of the skilled user. Dynamic time warping is used to ensure that the different samples match up.

2. Related Work

Classification of human activities is a well-known machine learning problem. Work in [Brand et al., 1997] used HMM to classify complex activity. In this work, it is argued that simple HMM has the critical limitation of assuming the system, as a single process with a few states, therefore conventional HMM is not suitable for such detections. Therefore couple HMM is introduced in the work. [Liu et al., 2010] used accelerometer for gesture classification. In this method, raw input is quantitated before be used for training an HMM to classify gestures. An extension of human activity detection is to detect an activity in a stream. A study of Chambers et. al. in 2004 conducts a set of experiment to explorer and compare performance of different time series representations and distance measures [Chambers et al., 2004]. Later in 2008 Ding et. al. tried to segment sport activities using time series analysis. In their work they used HMM on a stream of accelerometer data to detect sport activities [Ding et al., 2008].

3. Motion Capture Process

We have recorded 8 different activities (Table 1). For each activity we did recordings on two different positions on the body: the arm (armband), and wrist (wristband). We used our right hand and right arm since we collected data from users that are right-handed. For each setting, 5 recordings were done to generate a total of 80 recordings. Since we currently do not have any professional trainers or athletes to obtain good data from, we recorded data using the partner who we felt was the most proficient at each activity.

Recordings collect a time series of various features from iPhone 6 sensors (Table **??**)) at a rate of 100 frames per second. We used a smart phone to stream the data from the accelerometer through Wi-Fi to a computer where it was logged in a .csv file. We converted these .csv files to .mat files so we could analyze them in MATLAB.

Table 1: List of Collected Activities

Sport	**Activity**
Exercise	Squat
	Press
Basketball	Jump Shot
	One-Hand Pass
	Two-Hand Pass
Tennis	Forehand
	Backhand
	Serve
Ping Pong	Forehand
	Backhand
	Serve

4. Method

In order to determine if an action is performed correctly, we first need to identify the type of action. For that we first need to know what sport is being played at that moment and then search for the most likely action. It might also be possible to recognize an action without knowing the kind of sport, however this makes the problem harder and will increase the response time. In addition we assume that when someone is playing a sport he or she will not change to another sport, and it should be convenient enough to ask the user to at least choose the kind of sport.

To recognize the actions we build a classifier for actions of each sport. The classifier is made with a Hidden Markov Model (HMM) for each action and recognizes a query action by comparing the likelihood of models. HMM assumes that a hidden process controls the probability of observing an event [Rabiner and Juang, 1986]. The process has different hidden states and each state models a different distribution of data. In addition, the Markov property implies that the probability of being at a state at time t only depends on the state of the model at time $t-1$. The transitions of hidden states can have an ergodic architecture where the future state of the model could be any of the $\{s_1, s_2, ..., s_N\}$ states. Also we may use a left-right architecture and only allow auto-transition, and transition to the states with index greater than the current state.

In this work we use a left-right model that only allows auto-transition and transition to the next state. This is convenient for recognizing sequences that follow a similar pattern and enforces the model to learn a continuous pattern over the action. We model the actions using HMMs with 4 hidden states. Each state models the movement features with a mixture of 2 Gaussians.

We use the data from a gyroscope and accelerometer to construct the movement features. The gyroscope measures the rotation over the three axes which is a good feature for measuring the curvature of movement. The accelerometer measures the acceleration that is relative to the gravity, which is slightly different from the kinematic acceleration. An accelerometer in the rest position outputs 9.8 on the axis that is flat to the ground, which is the gravity force. This will cause a low frequency on the axes of accelerometer that are pointed to the ground. By taking the derivative of acceleration, which is jerk, we can also filter out the low frequencies.

To recognize actions in real-time we need to consider different factors. First, the model should have a considerably low process time. Also we should be able to recognize actions without having information about the beginning and end frames of an action. Since we use a left-right architecture for the HMMs, the processing time is linear and it needs NT computations to identify the likelihood of an input action, where N is the number of states of model and T is the length of sequence. This is much faster than using an ergodic model which needs N^2T processes.

We use a sliding window approach to recognize actions in real-time. With this method we segment the input data into parts with equal lengths of 32 with 31 frames overlap. For training the HMMs we use no overlap to avoid redundant samples. Since the data sampling rate is 100 frames per second 32 frames means about $\frac{1}{3}$ of a second delay in output, which for our purposes is a good response time.

After the gesture recognition process, we need to identify the parts of an action that are not accurate. For this reason we proposed to find the similarity between every frame of the action to a good sample of that action we have in the database. We use Dynamic Time Warping (DTW) to find the distance between the frames of a query action and the good sample of that action. With DTW we can find the closest match of a query frame on the sample in our dataset, and find the distance between the features at that point. For this analysis we use only the rotational features.

To identify the inaccurate parts with DTW, we need to have an entire sequence of an action. Therefore we should be able to determine an action from the transition parts between the consecutive actions and identify the beginnings and ends, respectively. For that we simplify the idea in [Lee and Kim, 1999] and use an additional HMM to recognize non-action parts. We call this HMM the threshold model. We train the threshold HMM using all the samples of actions we have for a sport. This model should output a likelihood less than the likelihood of the action classifier for an action gesture. This is because the parameters of threshold HMM are trained on all the actions and it should be less certain than the correct model of an action. On the other hand, the threshold model will output a likelihood higher than the other models on non-action sequences because of the large variance of its training data.

5. Results

In this section we discuss the results of classification and error recognition of actions in table tennis. We trained an HMM for forehand, backhand, and serve actions using 10 samples in each dataset. Figure 1 shows the 6 acceleration and rotation features and the most likely state sequence of each action using a sample in our test data.

By looking at the most likely hidden sequences we can tell that the classifiers divided the actions into 4 parts, where the beginning part is when the hand starts from rest. The second part is the preparation phase where the racket goes slightly back to prepare for hitting the ball. The third part is when it is moving towards the ball and the fourth part is when the hand goes back to rest. As we can see, the preparation phase in forehand and serve is longer than backhand. The reason is because in the backhand gesture the hand does not have enough space to prepare for hitting the ball, while in forehand and serve we have much more space.

Table 3 shows the F1 score of recognizing the three gestures. Using the entire length of gestures for classification resulted in perfect classification. Using a sliding window also we had perfect recognition of the backhand and about 90% accuracy in recognizing forehand and serve actions.

In a different experiment we also included the threshold HMM in the classifier to identify non-action parts. Table 4 shows the precision, recall, and F1 score of this test. As you can see, the precision of recognizing backhand and serve is 1, but for forehand and non-action gestures we received about 80% precision. On the other hand the recall rate for non-action movement is 1. With this information we can tell that the classifier can perfectly recognize all the transition parts, but it sometimes classifies a non-action movement as forehand.

One problem novice players have when starting table tennis is with the serve action. To have a good serve one should control the movement of the wrist and

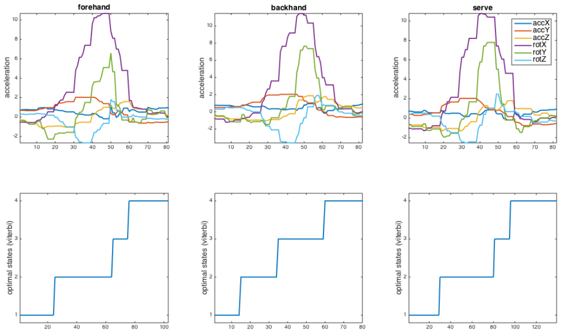


Figure 1: Most likely state sequence of a serve test action using Viterbi

prevent it from bending. We collected examples of bending the hand when serving and experimented to see if we can recognize this error with DTW. Figure 3 shows the distance matrix and the corresponding frames between a bad sample of the serve action and the good sample in our dataset. Also figure 2 shows the comparison of the error rate of a bad sample and a good sample. As you can see on the bad serve sample, we get a very high error when the player hits the ball because of bending of the wrist.

	Forehand	Backhand	Serve
entire sequence	1.00	1.00	1.00
window size 16	0.86	0.89	0.84
window size 32	0.91	1.00	0.93

Table 3: F1 score for recognizing each action using the whole gesture and sliding window of size 16 and 32 frames

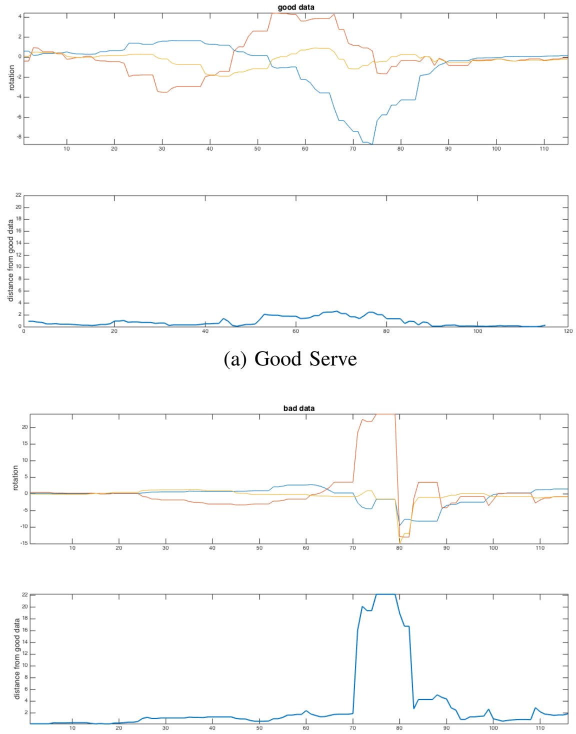
	Forehand	Backhand	Serve	Non-Gesture
Precision	0.84	1.00	1.00	0.86
Recall	0.92	0.77	0.78	1.00
F1 score	0.88	0.87	0.87	0.92

Table 4: Recognizing ping pong gestures from transition parts using a sliding window of size 32

6. Discussion and Future Work

The goal of this system is to be able to run in real-time and help the user to understand how good he or she performed an action. It also should provide a score to the accuracy of the entire session to help the user understand how much he or she has improved. In addition, we would like to provide a summary about the accuracy of performing each action and the parts of action that have the highest error rate.

In order to notify the user about the inaccurate parts in his or her performance we proposed that we



(a) Good Serve

(b) Bad Serve

Figure 2: 3D rotation and error rate of a good and bad serve computed with DTW

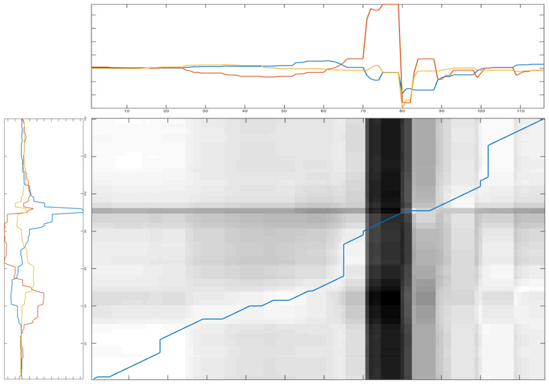


Figure 3: Distance between the 3 axes rotation of a good and bad serve

can vibrate the cell phone when there is an error in some part of the action. Another possible approach is to show the places of error on the movement path. However we cannot measure the position using the phone sensors. A possible approach for constructing the movement trajectory from the mobile sensors is to use an Extended Kalman filter. [Sabatini, 2006] proposed a method with Extended Kalman filters that can recognize the position using the accelerometer, gyroscope, and magnetometer. The proposed filter aims to not only determine the position via an orientation quaternion, but also the accelerometer and magnetometer bias. Understanding the bias from the accelerometer as well as the orientation quaternion can help us better understand what the true movement of the user is like.

Another issue of the current system is with the DTW algorithm. DTW needs T^2 computations, where T is the length of a query gesture. This might cause problems when the length of a gesture is long. [Salvador and Chan, 2007] proposed a modified version of this algorithm that is able to process in T computations and claimed that it is more accurate than the other methods for DTW. The FastDTW computes a subsample of a sequence with length $\sqrt{T}$ and by limiting the search space computes the distance in a time less than T. Then it projects the estimated path from the subsampled sequence on the entire sequence and tries to estimate the correct points within a radius of frames around the estimated path.

7. Conclusion

We presented a system for real-time classification of sport gestures and identifying the inaccurate parts of the gestures. Our classification and error estimation process showed promising results in identifying the correct action and the erroneous frames of action.

References

[Brand et al., 1997] Brand, M., Oliver, N., and Pentland, A. (1997). Coupled hidden Markov models for complex action recognition.

[Chambers et al., 2004] Chambers, G. S., Venkatesh, S., West, G. A. W., and Bui, H. H. (2004). Segmentation of Intentional Human Gestures for Sports Video Annotation.

[Ding et al., 2008] Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. (2008). Querying and mining of time series data: experimental comparison of representations and distance measures. pages 1542–1552.

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

[Liu et al., 2010] Liu, J., Pan, Z., and Xiangcheng, L. (2010). An accelerometer-based gesture recognition algorithm and its application for 3D interaction. *Computer Science and Information ...*.

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

[Sabatini, 2006] Sabatini, A. M. (2006). Quaternion-based extended kalman filter for determining orientation by inertial and magnetic sensing. *Biomedical Engineering, IEEE Transactions on*, 53(7):1346–1356.

[Salvador and Chan, 2007] Salvador, S. and Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580.

Table 2: List of collected features in each recording

Category	**Feature**	**Description**
General	Time	Time at the beginning of the recording.
	Time Stamp	Time as a long integer.
	Record Time	Time since recording.
GPS	Coordination {lat, long}	The geographical coordinate information.
	Altitude	The altitude measured in meters.
	Speed	The instantaneous speed of the device in meters per second.
	Course	The direction in which the device is traveling.
	Vertical Accuracy	The radius of uncertainty for the location, measured in meters.
	Horizontal Accuracy	The radius of uncertainty for the location, measured in meters.
	Local Time Stamp	The time at which this location was determined.
Accelerometer	Acceleration {X,Y,Z}	The acceleration value for the {x,y,z} axis of the device.
Magnetometer	Heading {X,Y,Z}	The geomagnetic data (measured in microteslas) for the {x,y,z}-axis.
	True Heading	The heading (measured in degrees) relative to true north.
	Magnetic Heading	The heading (measured in degrees) relative to magnetic north.
	Heading Accuracy	The maximum deviation (measured in degrees) between the reported heading and the true geomagnetic heading.
Gyroscope	Motion Rotation Rate {X,Y,Z}	The rotation rate as measured by the devices gyroscope.
Pedometer	Activity	Getting the Type of Activity: stationary, walking, running, automotive, cycling, or unknown.
	Activity Confidence	The confidence that the motion data is accurate.
	Activity Start	The time at which the change in motion occurred.
	Steps	The number of steps taken by the user.
	Last Step Date	The time at which steps are counted.
Processed Data	Rotation {X,Y,Z}	The rotation rate for each of the three axes in radians per second
	Yaw	The yaw of the device, in radians.
	Roll	The roll of the device, in radians.
	Pitch	The pitch of the device, in radians.
	User Acceleration {X,Y,Z}	The acceleration that the user is giving to the device.
	Device Orientation	The way user holds the device