Activity Recognition using Mobile Smartphones

Abdullah Alhoshan, Nicholas Jewell, Zhen Hu, Zhe Zhang
University of Nebraska-Lincoln

Abstract

During the last decades, smart phones have been dramatically changing the life of human beings. It becomes multi-functional device in our daily life, and people starts thinking to use it to keep track of activities on real time. In this paper, we propose two models to estimate human activities. First, we use a model of sliding window and evaluate different classifiers to determine the activity within each window. Second, we apply hidden markov model to directly handle sequential data and predict the state sequence based on our observed data. With these two type of models, we find that decision tree based classifiers give us very high accuracy to estimate activity in each window. Additionally, except the ambiguity introduced by two activities: stand and sit, most of the activities can be correctly estimated by hidden markov model.

1. Introduction

As smart phones become increasingly more necessary in our everyday lives these mobile devices have many built in sensors that can be used to obtain useful information such as: gyroscopes, accelerometers, GPS, cameras and so on. Data can be gathered from these sensors to determine what the user is doing. Many smart devices can already do this but in a more general scope. These phones come with fitness apps that can detect when the user is running, walking, biking, and more. So for this project, we want to identify the common activities people are doing in their daily life. Here are the three objectives of our project:

- To detect five basic activities: sitting/standing, walking up and down stairs, and running by using a Hiden Markov Model (HMM).
- To predict more human activities by including Gyroscope data.
- To generalize huamn behavior habit pattern by comparing different models with HMM .

To accomplish the above three objects, we will first transform the data from a temporal domain to a frequency domain and extract features from this transformed data. From there we can use Weka to classify these activities for this one frame. Finally, we can then use sequential data to create a Hidden Markov Model (HMM) to compare to the Weka results. The main contribution of our project is

The remaining of this paper is organized as follows: Section 3 provides background information Section 2 discusses related work. Section 4 describes our dataset and how we preprocess it. Section 5 describes our implementation details. Section 6 explains about results for each object. Section 7 concludes our final objectives.

2. Related Work

For human activity recognition and pattern discovery, lots of research has been done in this field. T. Gu [2] did the study about recognizing concurrent activities. Concurrent activities are those activities

people conduct at the same time. For example, people can watch television while eating food. People can walk while talking to their friends. These behaviors should be recognized using a different approach from that for sequential activity. Some other studies are about interleaved activities. Interleaved activities are those activities people can perform in different time period [2]. For instance, while cooking, if there is a call from a friend, people pause cooking for a while and after talking to their friend, they come back to the kitchen and continue to cook. For this interleaved activities, The Hidden Markov Model are widely used. HMM model turns out to be works very well in predicting basic human behavior [3]. According to the study by Andrea Mannini and Angelo Maria Sabatini, They used a Fourier Transformation [4] to change the data gathered from a temporal domain into a frequency domain. By applying HMM, they could successfully use the data gathered from accelerometers to detect the activity the user is currently performing.

3. Background

We provide some background knowledge in this section. We only give a brief introduction of the models and techniques we use in our project.

3.1. Fourier Transform

The Fourier transform is important in mathematics, engineering, and the physical sciences. Its discrete counterpart, the Discrete Fourier Transform (DFT), which is normally computed using the so-called Fast Fourier Transform (FFT), has revolutionized modern society, as it is ubiquitous in digital electronics and signal processing.

The Fourier transform is a reversible, linear transform with many important properties. For any function $x(j)$, the Fourier transform can be denoted $X(k)$. Often $x(j)$ is a measure of time j (i.e., the time-domain signal) and so $X(k)$ corresponds to inverse time, or frequency (i.e., the frequency-domain signal).

In this project, we use FFT to transform our data from time domain to frequency domain. This give us more comprehensive view of our data and enrich the scope of features to mining our dataset.

3.2. Hidden Markove Model

In our project, we will use Hidden Markov Models (HMM). HMM is a generative probabilistic model, which is a model that is used for generating hidden states from observable data [2].

Hidden Markov Model(HMM) is an common model in various applications such as pattern recognition. It was heavily applied to text mining. We need three parameters to define our model: Initial Probability, Transition Probability and Emission Probability. Initial probability represents the probabilities in each initial state and how they transit between each other. Transition probability indicates how much chance each state transform to other state. Emission probability provide the possible outputs given the current state. With these three parameter, we could predict the output sequence which maximize the posterior joint probability between input sequence and output sequence. Let's summary the model with the following equation.

$$P(x_1, x_2, x_3, ..., x_n, y_1, y_2, y_3, ..., y_n) = P(y_1)P(x_1|y_1)\prod_{k=2}^{n} P(y_k|y_{k-1})P(x_k|y_k) \tag{1}$$

In Equation 1, the product of $P(y_1)$ and $P(x_1|y_1)$ gives the joint probability of the initial state. $P(y_k|y_{k-1})$ implies the probability of transmitting from state y_{k-1} to state y_k. This series of $y_1, ..., y_n$ are the transition probabilities. It is usually represented by a matrix and the element in the entry i, j is the probability from state i to state j. Finally, $P(x_k|y_k)$ is the emission probability in state y_k. It is the probability of emitting x_k given the current state of y_k. The goal of HMM is to compute state sequence $\overrightarrow{y}$ so that given input sequence $\overrightarrow{x}$, the joint probability of $P(\overrightarrow{x}, \overrightarrow{y})$ is maximized. Figure 1 shows a graphical representation of an HMM [5].

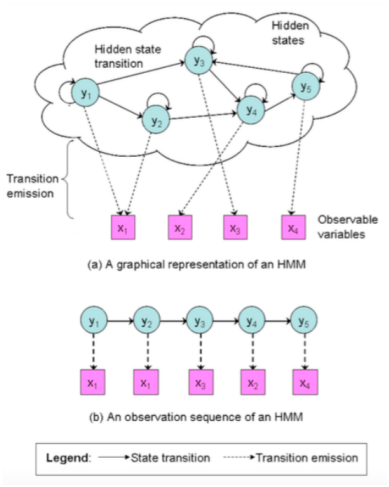


Figure 1. Hidden Markov Model

4. Methodology

All data are collected with an application on Apple Store called SensorLog. It record a sample every 33 ms. The dataset is gathered from each member of the project team. Each member is responsible for recording each activity ten times. This gives forty records for each activity. We further divide our data into two subsets: training data and test data. We randomly pick one record out of ten from each person and put 4 records into test set. The rest of 36 records are training data. We training our model in Weka with training data and provide test data to evaluate the accuracy of our model. SensorLog is able to export the records to a csv file. Each instance of record contains 57 attributes. We extract the accelerometer data from raw csv file. There are three directions X, Y, and Z therefore three attributes of the data are extracted.

4.1. Preprocessing

Since all of our data are represented in time domain. To enrich our dataset, we transform our data from time domain to frequency domain with Fourier Transform. We believe this procedure is necessary to differentiate some activities.

4.2. Feature Extraction

With Fourier Transform, we have data in both time domain and frequency domain. We calculate four features in each dimension and they are:

- Digital Component (DC)
- Mean value in time domain
- Entropy in frequency domain exclude DC component
- Power

Digital component is the data with frequency zero. Since in Z-axis, there is always an effect due to gravity. We separate DC from other frequencies in order to get rid of the gravity effect in Z-axis. Additionally, DC component is the dominant data when entropy and power get computed in frequency domain. Reducing the effect of DC in frequency domain will give us more resolution of frequency data. Finally, we have four features in each dimension X,Y,Z and hence 12 features in total.

4.3. Data Frame

Since we are dealing with sequential data, restricted by the memory space and CPU capability, it is impossible to process all the data at once. As data keeps coming in, we divide data stream into reasonable chunks. In other words, we could think our model as a sliding window. The window moving toward right and we only need to memorize the latest samples. Figure 2 illustrate our models. We run some measurements and find that 32 samples are sufficient to get high accuracy for our classifiers. We will discuss more details in next sections. 32 samples is relatively 1 second. To reduce data size, we extract single instance with 12 features out from one frame. Furthermore, two data frames are overlapped with each by half of the frame. Therefore, we obtain each instance every other 16 samples and consequently drop our dataset by 16 times.

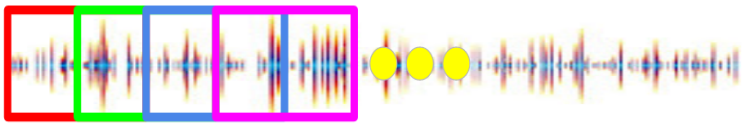


Figure 2. Sliding Window

5. Implementation

We will use a a variety of tools to complete our objectives. We will mainly use a two step process to get our results. First we will use a Fourier Transformation, and then we will create an HMM. We will also use Data Mining techniques in Weka.

5.1. Fourier Transformation

We use python library scipy and adopt fft functions from the package fftpack. Since the sampling rate in SensorLog is 33 Hz, our FFT contains the frequency between -16.5 Hz to 16.5 Hz. Figure 4b shows an example of running activity. The top three figures illustrates the data in frequency domain and the bottom three figures shows the data in time domain. We exclude DC component from the figure therefore there is no data appears in frequency zero. Due to the symmetric property of FFT, we only keep positive frequencies in FFT.

After we calculate FFT, we need to compute DC component, Time Mean, Entropy and Power. We have already discussed philosophy behind these features in previous sections. Before writing equations of different features, we need to briefly talk about Power Spectral Density (PSD) since we need to use it to calculate frequency entropy and power. PSD is simply squaring the amplitude spectrum and scaling it by number of frequency bins like Equation 2 and 3.

$$P(\omega_i) = \frac{1}{N}|X(\omega_i)| \tag{2}$$

$$p_i = \frac{P_i}{\sum_i P_i} \tag{3}$$

Since we find our amplitude from FFT is very low and can not be represented in a good resolution, we ignore the normalization step in Equation 2 and keep the total PSD for each window. With the knowledge of PSD, now we are ready to list how we calculate these features:

- DC Component $x_{DC} = X[0]$, where X is the output sequence of FFT
- Time Mean $X_{mean} = \sum_{k=0}^{31} x_k$, where x_k is the time samples within a window
- Power $P(\omega_i) = |X(\omega_i)|$
- Entropy without DC $H = -p_i \log p_i$

We implemented above functions to extract 12 features from raw data file and import features into Weka for analysis.

5.2. Hidden Markov Model

We use a library called hmmlearn to implement our model. It is a unsupervised learning model and it is further categorized into several sub-models based on the internal soft-hard decision techniques. hmmlearn contains three models: MultinomialHMM, GaussianHMM, GMMHMM. MultinomialHMM is only able to handle nominal data and we consider it not suitable for our dataset. GaussianHMM is using single dimension of Gaussian random variable to estimate the input sequence. GMMHMM is an extension of GaussianHMM and use Gaussian Mixture Model (GMM) to estimate the sequence. The difference between GaussianHMM and GMMHMM is that GMM uses multiple Gaussian random variables to estimate input sequence and it should have higher accuracy than GaussianHMM. Unfortunately, GMMHMM contains serious bugs in the library when we implement our model, so we use GaussianHMM in our project.

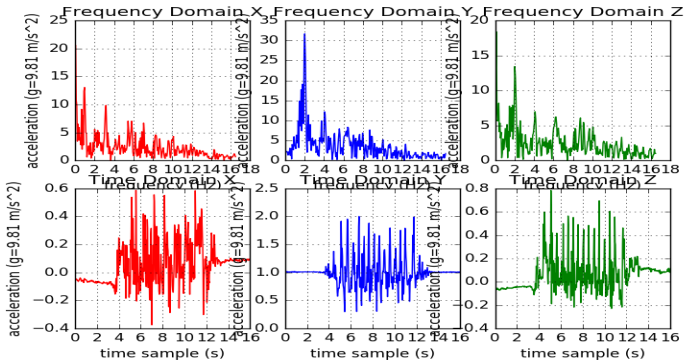
The interface of function is very friendly to use. We need to input the number of states. In our case, we examine 5 activities so 5 states. Then we input our training data sequence to train the model. In the end, we can get the transition probability matrix and emission probability matrix. The hmmlearn choose default initial probabilities if they are missing. After getting these parameters, we can predict states sequence with the model. We split our data set into training data and test data and run our model with both datasets to validate our results.

6. Results

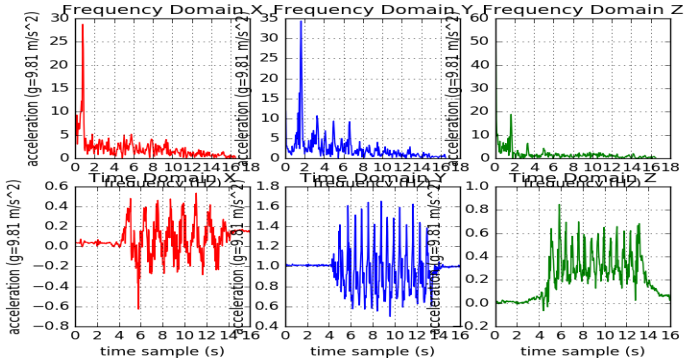
6.1. Fourier Transformation

We calculate Fourier Transform for each activity but due to the limited space of the paper, we only shows some interesting results here. These results reveal the patterns of our raw data in both time domain and frequency domain. Consequently these help us to carefully design features to distinguish different activities.

6.1.1. Downstairs VS. Upstairs. Figure 3 shows the data of walking downstairs and upstairs. If we give a glance at both time domain and frequency domain. Their patterns look very similar. The FFT of these two activities are illustrated in different scale at y-axis. But if we further examine the time domain at Z-dimension. We will find that the have different average values. Walking downstairs has the average value of 0.1 in time domain but walking upstairs has the average of value 0.4. Therefore, with the feature of mean value in time domain, we should be able to distinguish between these two activities.



(a) Walking Downstairs

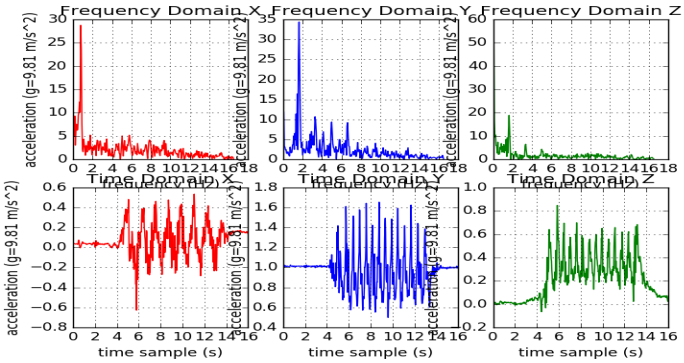


(b) Walking Upstairs

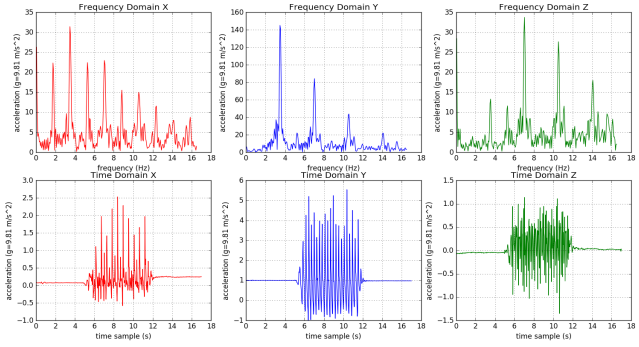
Figure 3. Comparison between walking downstairs and upstairs

6.1.2. Upstairs VS. Run. Figure 4 compares the data between walking upstairs and running. We can see the data in frequency domain looks very different from each other. Therefore, we choose the entropy

as our feature since it measures the uncertainty of the data. If we take a look at Figure 4b, we could expect higher entropy since the FFT of running is more disturbed.



(a) Walking Upstairs



(b) Walking Upstairs

Figure 4. Comparison between walking upstairs and running

We do not demonstrate other FFTs. But we do examine the data from every activity and each team member and eventually decide the features we are using in the project.

6.2. Single Frame Analysis

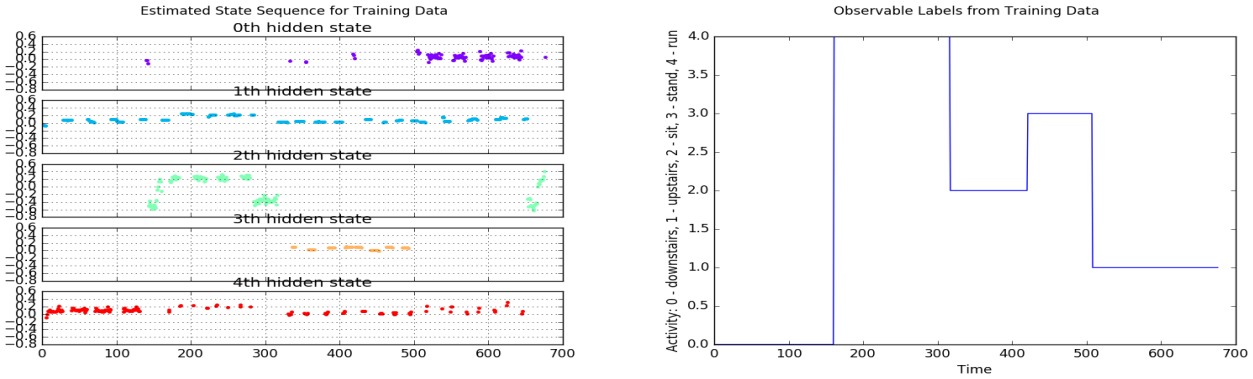
We import 12 features into Weka and run different models to classify our activities. We choose 4 classifiers: Naive Bayes, J48, Bagging and Random Forest. The result of 4 classifiers shows in Table 6.2. The accuracy of Random Forest model surprise us and all the samples are predicted correctly. Our original purpose of doing single frame analysis is to improve the accuracy of each data frame. We believe accurately predicting time sequence with in single frame will help us in HMM. But as we see, Random Forest give us very high accuracy, so we should be confident to move to HMM model and analyze the sequential data. We need to point out that our results only apply to our team members and five activities. If the subjects and activities change, the accuracy of the model might vary.

6.3. Hidden Markov Model

Figure 5 shows the output from our HMM model. x-axis is the time series of data samples. y-axis is the post-probability of the state. We need to aware that at each time moment each state will

Model Name	Correctly Classified	Incorrectly Classified	Root Squared Mean Error
Naive Bayes	48.37	51.63	0.3714
J48	94.77	5.23	0.1299
Bagging	93.46	6.54	0.1628
Random Forest	100.00	0.00	0.0705

emit a estimated output with certain probability. hmmlearn choose the state with the most probability. Therefore, at each time point in x-axis only the most competitive state appears in Figure 5. We phrase five states in Figure 5a as state $0, 1, 2, 3, 4$ in the following sections.



(a) Estimated Output Sequence of HMM

(b) Actual Activity of Input Data

Figure 5. Comparison between Estimated Activities and Real Activities of Training Dataset

In Figure 5, we found several interesting things.

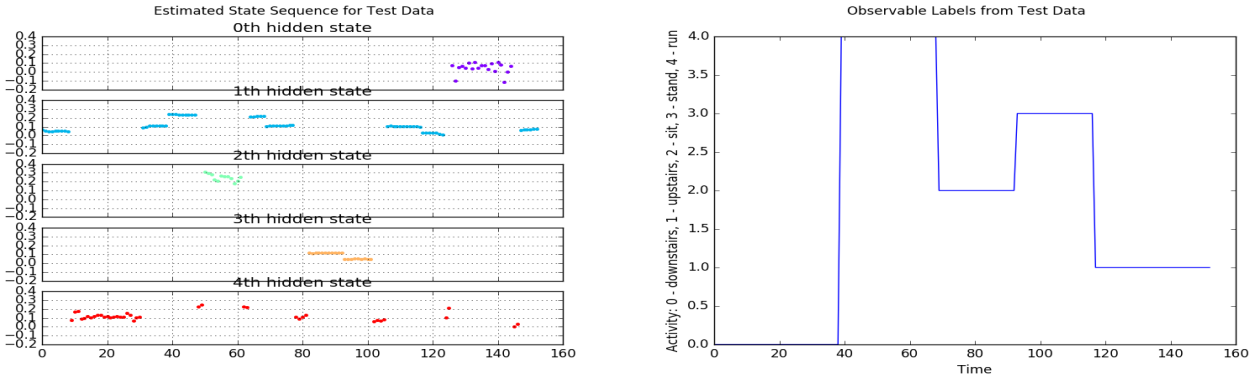
6.3.1. Noise in collected data. Let's take a look at the state 2. Most of the time series this state appears match the actual activity of running. Therefore, this state very likely stands for activity of running. However, if we examine the time samples between 140 and 320 into more depth, the post-probabilities during this period follow a shape of parachute: it starts increasing at the first half of samples and decreases at the second half. The reason is due to the noise in our collected data. Let's memorize the procedure of collecting running data. We starts SensorLog and then run get started to run and stops SensorLog. There a slack time period after we starts the app and before the activity actually happens, but this period of data is recorded in SensorLog. The same noise applies to the end of the activity. If we look back to Figure 5a, HMM is not quite sure at the beginning of the running due to the noise (we starts the app but not running yet). Later on, as we starts running, the confidence becomes high to emit the output as state 2. The same analysis should be applied to the second half of data samples of running.

6.3.2. Walking downstairs is easy to be erroneous estimated. If we look at Figure 5a, the most possible state for walking downstairs is state 4 since it has the most of time samples match the actually activity. However, it has a very long tail and lasts until the end of all time samples. This indicates walking downstairs might have a lot of features very similar with rest of the activities. To distinguish this activity from others, we need to add more unique features for walking downstairs.

6.3.3. It is difficult to distinguish between stand and sit. From the state 3 in Figure 5a, it appears during both stand and sit. This ambiguity makes sense because both time domain and frequency domain

look very similar for those activities. To further separate these two activities, we could add a feature and calculate discrete derivative of input sequence. Our hypothesis is that this feature should be completely opposite between these two activities and it should be very easy to distinguish them.

In order to validate our results, we use our test dataset and rerun HMM model. The output shows in Figure 6.



(a) Estimated Output Sequence of HMM

(b) Actual Activity of Input Data

Figure 6. Comparison between Estimated Activities and Real Activities of Test Dataset

We can see the results look similar as training dataset. Therefore, we consider our model should be highly consistent in different dataset. Please note here we only collected data from our team members with five activities. To further validate our model, we should try different subjects and different activities.

7. Conclusion

In this paper, we talked about how to use data from accelerometers to estimate human activities. From the experiments, we get some important results. First, we use different classifiers to classify our data. We found that common decision tree based classifier give us very high accuracy. This points a direction for future research on this field. We could embed decision tree classifier into a sliding window and estimate the activity of the data in each window frame. Second, we also implement hidden markov model and directly handle the sequential data. Although data noise has some bad impact on our posterior probability in HMM, the overall estimation is acceptable. Two activities need investigation in more depth: stand and sit. They are ambiguous in HMM model because the features are very similar. We expect to add new feature such as derivative of time series in future and it is interesting to see if it can help us to distinguish them.

Acknowledgments

The authors would like to thank Dr. Samal for his insightful recommendations on our projects as well as his effort of introducing us to this domain. In addition, We also appreciate Natasha Pavlovikj's tremendous help in this course. Finally, other groups have excellent projects and some of the ideas in our project are inspired from their work.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[2] T. Gu, Z. Wu, X. Tao, H. K. Pung, and J. Lu, epSICAR: An Emerging Patterns based approach to sequential, interleaved and Concurrent Activity Recognition. Proceedings of the IEEE International Conference on Pervasive Computing and Communications, 2009.

[3] Mannini, Andrea, and Angelo Maria Sabatini. "Machine Learning Methods for Classifying Human Physical Activity from On-Body Accelerometers."Http://www.mdpi.com/1424-8220/10/2/1154/htm. Sensors, n.d. Web.

[4] Morris, Dan. "An Intuitive but Not-All-That-Mathematically-Sound Explanation of the Fourier Transform." An Intuitive but Not-All-That-Mathematically-Sound Explanation of the Fourier Transform (n.d.): n. pag. Web.

[5] C. Sutton and A.McCallum, An introduction to conditional random fields for relational learning, In L. Getoor and B. Taskar, editors, Introduction to Statistical Relational Learning. MIT Press, 2006.

[6] Eunju Kim, Sumi Helal, and Diane Cook. 2010. Human Activity Recognition and Pattern Discovery. IEEE Pervasive Computing 9, 1 (January 2010), 48-53. DOI=http://dx.doi.org/10.1109/MPRV.2010.7

[7] Sensor Log. https://itunes.apple.com/us/app/sensorlog/id388014573?mt=8

[8] "HMMWeka." HMMWeka. Web. 09 Mar. 2016.