Biometric in Motion: Identification Using Acceleration Data

Zexi Mao
zexim@andrew.cmu.edu

Siqi Tan
siqitan@andrew.cmu.edu

Yuchen Wu
yuchenw@cs.cmu.edu

Yimeng Zhang
yimengzh@cmu.edu

Abstract

As cellphones and other smart devices are closer to our daily life than ever, we are curious to know how the data gathered by accelerometers on them impact our lives, particularly in terms of identifying users of devices via such data. In this project, we try to answer this question with machine learning techniques. We will first study how to implement such biometric techniques by retrieving useful identification information from raw accelerometer data and building classifiers. Then we will show how such techniques impact our lives.

1 Introduction

1.1 Motivation

Why is it important Ubiquitous computing had never been so close to us as our phones become smarter and more powerful today. Sensors equipped in our cellphones like GPS, gyroscope, accelerometer and even barometer collect data from the surrounding environment and ourselves. This information collected from our cellphones and other mobile devices can boost many more applications than just adjusting the brightness of the screen and rotating it automatically. Google uses the GPS data from our phones to provide real-time traffic conditions. Apps can track how well you sleep by just putting your phone beside you on bed. Indoor localization takes advantage of wireless fingerprint, sound from microphone, footsteps from accelerometer and even colors from the camera to tell you where you are when GPS signal is weak inside buildings. Fancier applications far beyond the original purposes of these sensors are on the way. How to retrieve more information from the raw sensor data is one of the hottest and promising topics today.

What is the problem Accelerometer is one of the most interesting sensors in our cellphone, which records the acceleration data in 3D. Posture of the phone, gestures and footsteps of the user, and even the user's sleeping condition can be measured by the accelerometer.

In this project, we tried to explore a novel usage of the accelerometer: biometric identification. In other words, can we identify the user by only looking into how he (she) moves? We believe that everyone has his (her) own unique pattern of movement. If this assumption is true, we can identify the person when we match the current data from the accelerometer with the historical data we have learned.

Brief outline of solution In this project, we chose the already available accelerometer dataset gathered by Kaggle. Due to the poor structure of the raw data set, we first did some preprocessing, mainly splitting and re-sampling. Then we extracted discriminative features from preprocessed dataset. The features we used included FFT coefficients, the distribution of timestamp intervals, and the set of possible acceleration readings. Finally, we applied several classifiers to the feature vectors. Now we

have got a score of almost 0.90 in terms of area under the ROC curve, which is satisfactory, although further improvement is still needed.

1.2 Related Work

haven't checked yet...

A human beings walking gait can reflect the walkers physical characteristics and psychological state, and therefore the features of gait can be employed for individual recognition. The use of accelerometer data for biometric identification is relatively new but has been increasingly explored in recent years. Existing methods for gait recognition have shown good performance.

Gafurov et al. [1] attach multiple sensors to a subject at different body parts. Xu et al. [2] developed an Android App to collect gait acceleration data. With a reasonably sized dataset, by matching gait patterns across different paces, they show preliminary results indicating that not only can smart phones be used to identify a person based on their normal gait but also that there is potential to match gait patterns across different speeds.

Tao et al.[5] focus on the representation and pre-processing of appearance-based models for human gait sequences. Two major novel representation models are presented, namely, Gabor gait and tensor gait. Experiments show that the new algorithms achieve better recognition rates than previous algorithms.

Pan et al. [4] proposed algorithm based on signature points, instead of the whole gait signal. They consider acceleration-based gait recognition insensitive to changes of lighting conditions and viewpoint. Their algorithm firstly extracts signature points from gait acceleration signals, and then identifies the gait pattern using a signature point-based voting scheme. The experimental results shows the accelerometer-based gait biometrics is promising.

Kwapisz et al.[3] collect some data and also perform identification experiments . Based on the 600 raw accelerometer readings, they generated 43 features, which are variations of 6 basic features including average acceleration value, standard deviation, time between peaks and so on. They applied two classification techniques decision trees neural networks to classify and the identification performance turned out to be fairly good.

2 Methodology

In this section, we describe the methods we have used for preprocessing, feature extraction, and classification. This is not necessarily the final version of our approach.

2.1 The raw data

The data set acquired from Kaggle consists of 3 parts: a training dataset, a testing dataset and a question set. Each single sample in the training and testing dataset contains a time stamp in milliseconds, acceleration measurements in 3 dimensions, and an associated DeviceId. There is an Android application published on Google Play for such data collection. This application runs in background while reporting accelerometer data and the corresponding device id to a central server.

In the training dataset, there are 30 million samples, which are collected from 387 different devices as labeled. In the testing data set, there are also 30 million samples, but without DeviceId. These samples are demarcated into 90,000 sequences, each with a SequenceId. In the question set, for each SequenceId, there's a proposed DeviceId.

The challenge on Kaggle is to tell whether each sequence's proposed DeviceId is the sequence's real DeviceId. For each sequence, we're required to give a belief, in real number, about the credibility of the proposed DeviceId. Therefore,

For our study, we aim at a more general version of the problem of classification, which is telling the correct DeviceId for any given sequence of sample.
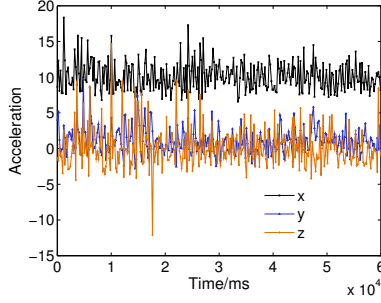
Figure 1: A good trace of raw data. Data is almost uniformly distributed along time.
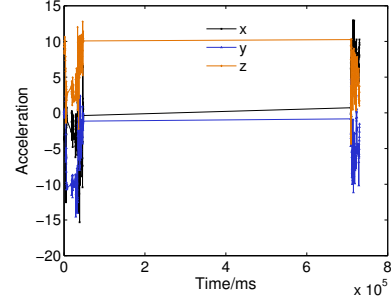

Figure 2: A bad trace of raw data. This piece essentially contains information about 2 separate activities that are far apart in time.

2.2 Refinement

The raw dateset has already been prepared preliminarily by Kaggle. First, the devices with fewer than 6000 samples are removed ahead. Second, sequences of samples during periods of rest exceeding 10 seconds are eliminated as well.

However, the dataset is still far from beautiful.

First, accelerometers have various sampling rates. A large interval between two samples may lead to much information loss between the two sample points. Furthermore, the acceleration on each dimension changes rapidly in such low sample rate. The high variance of sampling interval within one device's trace even makes it harder to retrieve features.

Second, according to the nature of Android system and the low privilege of an application, sometimes the app cannot access the accelerometer, sometimes the app itself is swapped out or killed by the operating system. Thus, a great part of the sample data is not well distributed over time.

Figures 1 and 2 show the two segments of untouched raw data.

To overcome the two challenges above. We propose our three phase refinement: splitting, smoothing and re-sampling.

The idea of splitting is very simple. For the first step of splitting, we scan a sequence of data from a identical device in chronological order. Once the interval between our current sample point and the previous one is greater than a threshold T_{split}, the samples scanned before are grouped into one sub-sequence. This process continues until all the data from each identical device is split completely. Then we go to the step two of splitting. In this step, if any sub-sequence is shorter than a threshold T_{short}, we simply eliminate it as we believe that a too short sequence brings more noise than useful information. If, unfortunately, all the data from a particular device is abandoned, it left us no choice but keeping the longest among its all sub-sequences.

The next phase is to make the sequence smoother, which hopefully could reduce the level of noise. To this end, interpolation is the idea that comes to us so far. We simply adopt cubic spline as our interpolating function. Thus, we transform a sequence of sample points into a continuous curve. Some missing pieces of data are "reproduced" in this way. This phase makes it easier for feature retrieval as we can now get the data at any time point. We should notice that noise is not necessarily reduced well in this manner. The curve we get contains almost the whole noise from the original data as it is with respect to every single data point from the raw data. We believe a better smoothing function will help while we are still looking for such one.

The third phase benefits from the smoothing phase as it is quite easy to re-sample data point from a smooth curve. We set a parameter, τ_{resample}, to be the re-sampling time interval. With this interval, a new sequence of data is generated from on the (interpolated) original data trace.

After the three phases, the refined data trace is produced from the raw data. Next, we will propose the way we select features from such refined trace.

2.3 Feature extraction

Feature selection as well as extraction is the most important part of the entire classification process. A great feature should pinpoint reveal the underlying differences between the users (also known as DeviceId in this project). In this section, we describe how to extract features from the preprocessed data.

The only assumption we made so far in this research project is that a human being moves in his/her own characteristic pattern. This pattern could be a sequence of outstanding events, or it could be some repeated, characteristic events as well. Noticing that accelerometer records data with unpredictable length and quality, we cannot assume that the complete sequence of events could be well recorded. Thus, our hope rests on the latter one.

It is customary to use frequency analysis for repeated events. In our approach so far, simple Fast Fourier Transformation is adopted. The basic idea is just making FFT on the refined data sequence we talked about in previous section.

A better idea is to make windowed FFT on them. The reason is, as each of the sequence has different length, if we make full FFT on each sequence, the frequency components produced by FFT have different meanings on different lengths of FFT. For example, we make full FFT on two sequence with length 17 and 29 respectively, the resultant points are not matched in the frequency domain. The results will be hard to be interpreted and compared. To this end, we make the transformation in a given window size, w_{fft}. Thus, multiple transformations are performed in a sliding window manner. We can average each corresponding points from different transformations, as they are matched in the frequency domain.

However, another concern arises. If we make one windowed FFT on a pretty long sequence, some information is erased by the sliding window averaging. This is true if we believe that a longer sequence could contain more information than a shorter one.

To this end, we prefer to perform partitioning on long sequences before doing windowed FFT. There are two thresholds in the partitioning process. One is the lower bound of number of generated sub-sequences, $k_{\#\text{seq}}$, the other is the upper bound of the maximum length among all the generated sub-sequences, l_{maxL}. The procedure of partitioning goes as follows. The input sequence is halved into two sub-sequences until either of the following conditions satisfied: a) the number of sub-sequences is greater than $k_{\#\text{seq}}$; b) all the sub-sequences are shorter than l_{maxL}. In this manner, on the one hand, we make more sequences from one device, hopefully keeping more information about the device's diverse patterns. On the other hand, we use the second threshold to make the sequences not too short, so that they contain reliable information about patterns of the device.

So far, the results of FFT is the extracted features as we shown above. As there are three dimensions, and the independent values of FFT with window size w_{fft} is $1/2w_{fft}$, we build a feature space with $3/2w_{fft}$ dimensions.

2.4 Classification

This is the ongoing part of out work. The basic idea is to leverage various kinds of classifiers and optimize their parameter for more accurate results.

We start from some naive classifiers. The performance of K-Nearest-Neighbor depends on the number of K and the definition of distance function. This classifier leads us to a baseline result. Then we move on to popular classifier such as support vector machine. Optimizing SVM via kernel and parameter selection will be discussed in next section.

3 Implementation and Evaluation

In this section we show how our approach performs. We will also manipulate and justify the selection of parameters of both reprocessing and classifiers.

3.1 Parameter selection

In this subsection we discuss how to determine the parameters and thresholds we defined in the previous sections.

T_{splite} is set to be 10 second in consistent with the preprocessing setting done by Kaggle. It is pointless to set the threshold above 10 seconds as those sequence shorter than 10 seconds have already been eliminated. We keep this threshold as large as possible to generate longer continues sequences. T_{short} is set as 4.8 seconds for training set and 3.2 seconds for test set based on empirical study.

We let $\tau_{resample}$ be 200 millisecond. This value is chosen based our statistic result of the distribution of the original data. Any value near 200 ms is a reasonable choice because most of the intervals lie in this range. Choosing a value deviating from 200 ms could misinterpret the data. (put a figure here for CDF of interval?)

The window size of FFT, w_{fft}, is 64 sample points, which means a window with length 12.8 seconds. We chose this value to balance the details of data and dimensions of feature space.

$k_{\#seq}$ and l_{maxL} are more flexible as the result is less sensitive on these values.

3.2 Optimizing classifier

This is the To Be Done part.

3.3 Results

Currently we are ranked at 250 out of 500 competitors on Kaggle.

Appendix

Our plan for future goes here.

If possible, we will evaluate our technique not by testing it on the Kaggle dataset, we will also develop apps for our smart phones to collect real time data for learning and evaluation. We hope this approach will give us the clue of how to protect our id privacy.

We will learn a few things from this project besides a better understanding of machine learning itself if our machine learning algorithms finally identify users. First, the assumption about pattern of movement could be true. Second, applications such as anti-theft, health monitoring and emergency detection that adopt this novel identification technique will be available in the near future. Third, we will be able to know how much data from the accelerometer is sufficient to cause the leak of one's identity, which can trigger a serious privacy issue.

References

[1] D. Gafurov, E. Snekkenes, and P. Bours, "Gait authentication and identification using wearable accelerometer sensor," in *Proceedings of IEEE Workshop on Automatic Identification Advanced Technologies*, June 2007, pp. 220–225.

[2] F. Juefei-Xu, C. Bhagavatula, A. Jaech, U. Prasad, and M. Savvides, "Gait-id on the move: pace independent human identication using cell phone accelerometer dynamics," in *Proceedings of IEEE 5th International Conference on Biometrics*, 2012, pp. 8–15.

[3] J. R. Kwapisz, G. M. Weiss, , and S. A. Moore, "Cell phone-based biometric identification," in *Proceedings of Fourth IEEE International Conference on Biometrics: Theory Applications and Systems*, September 2010, pp. 1–7.

[4] G. Pan, Y. Zhang, and Z. Wu, "Accelerometer-based gait recognition via voting by signature points," in *Electronics Letters, 45(22)*, 2009, pp. 1116–1118.

[5] D. Tao, X. Li, X. Wu, and S. Maybank, "General tensor discriminant analysis and gabor features for gait recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence ,29(10)*, October 2007, pp. 1700–1715.