smartphone Side-Channel Attacks and Defenses

**Module 3 lAB Manual**

**Lab Manual Development Institution:** Colorado School of Mines

**Lab Manual Contributors**:

Zhiju Yang\*, Weiping Pei\*, Ryan Jackson+, Chuan Yue@

(\*: PhD student and research/teaching assistant; +: Master Student; @: Faculty.)

# Motion Sensor Data Feature Extraction

**Lab Description:** In this lab, you will design and extract features for machine learning algorithms based on the preprocessed motion sensor data. You will write tools or code to derive both time domain and frequency domain features that will be later used to train the machine learning classifiers in the training phase, and also perform the input inference as well as user fingerprinting attacks in the attacking phase. We suggest you to use Python to write your tools or code so that the extracted features can be easily used in the follow up labs. This lab consists of six STEPs.

The high-level **learning outcomes** and the corresponding **assessment** of this lab are summarized as follows. In other words, upon completion of this lab, students should be able to:

* **Explain** the reasons for extracting statistical features from the preprocessed motion sensor data.
  + Assessed by the tasks and outputs specified in STEP 1.
* **Develop** code to extract simple statistical features in both the time domain and the frequency domain.
  + Assessed by the tasks and outputs specified in STEPs 2 and 3.
* **Explain** the relationship between time domain features and frequency domain features.
  + Assessed by the tasks and outputs specified in STEP 3.
* **Develop** code to extract advanced features such as the total energy statistical features.
  + Assessed by the tasks and outputs specified in STEPs 4 and 5.
* **Develop** code to save the extracted features to a file.
  + Assessed by the tasks and outputs specified in STEP 6.

**Lab Environment:** Linux, Mac, or Windows.

**Lab Files that are Needed:** TheLab Manual file and the motion\_data.pkl file.

**Learning Setting:** This lab module is for students to complete outside the classroom, so it can be used in either face to face or online courses.

**Prerequisites:** Java or Python Programming, Basic Cybersecurity and Machine Learning knowledge and skills, Linux or Windows Systems, Computer Networks.

**Length of Completion:** 600 minutes.

**Level of Instruction:** Senior undergraduate students or graduate students in CS or related STEM programs. The lab exercise should be further simplified if it will be used for freshmen, sophomores, or none-CS major students.

**Interconnection with Other Labs:** This lab module is standalone by itself; however, if needed, an instructor can use the details in the course project manual and the other four lab manuals to provide additional hints to students.

**Assessment Guideline:** Students should follow the steps to answer all the questions. Based on the points assigned to each individual question, the instructor will grade each answer (together with the additional materials if specified for the question) in terms of its correctness (60%), clarity (20%), and concision (20%).

### **Lab Exercise/step 1 (Download and load the preprocessed motion sensor data)**

Download the file motion\_data.pkl for the preprocessed motion sensor data that was produced in Lab 2, and write code to load the data, e.g., by using the Python *pickle* module. This should result in a long list of 3-tuples in your code for the motion sensor data of 30 users.

The first element of each 3-tuple is a matrix with 9 rows where each row is one of the 9 sequences of sensor data, the second element is the key or digit (e.g., digit “5”) typed by a user, and the third element is the ID of the user who typed the key. Each 3-tuple has the following format:

([ (accX\_1, accX\_2, …),

(accY\_1, accY\_2, …),

…,

(accgZ\_1, accgZ\_1, ...)],

“5”,

UserID)

The 9 rows in the matrix are time series motion sensor data (with the corresponding timestamps removed), showing the acceleration forces in the x, y, and z directions, the rotation rates in the alpha, beta, and gamma directions, and the acceleration forces with gravity in the x, y, and z directions, respectively. In this lab, these 9 types of data are considered as raw features represented by MX, MY, MZ, rAlpha, rBeta, rGamma, MGX, MGY, and MGZ, respectively.

**Question 1**: What is your Python code for loading the data in the file motion\_data.pkl by using the Python pickle module?  
(Total score: 10 points. Grading rubric:  
100% points for correct and complete code;  
60% points for partially correct and partially complete code;  
30% points for partially correct or partially complete code.)

**Question 2**: The preprocessed data loaded from motion\_data.pkl are in the time domain, where each 3-tuple represents the time series discrete values of the raw features for a specific digit typing. To train a machine learning model for certain classification task such as either inferring the typed digit or inferring the user who typed the digit, we often need to extract and use some statistical features such as the minimum, maximum, and mean values from the discrete values of raw features. Please explain the potential reasons for extracting and using statistical features.  
(Total score: 10 points. Grading rubric:  
100% points for explaining three potential reasons;  
60% points for explaining two potential reasons;  
30% points for explaining one potential reason.)

### **LAB EXERCISE/STEP 2 (Extract Simple features in the time domain)**

For the data of each keystroke in motion\_data.pkl, you should create a feature vector. Simply, you can start by taking the minimum, maximum, and mean values of each of the 9 time series for each keystroke. To do this, you should write code that creates a new matrix where each row corresponds to each keystroke, and the columns are the min, max, and mean values for each time series, followed by the key pressed and the user ID in the last two columns. In other words, each row is a record consisting of 29 fields as shown below, where the first 27 fields constitute the feature vector while the last two fields are two labels of the record that indicate the digit pressed and the user ID:

['MX\_min\_time', 'MX\_max\_time', 'MX\_mean\_time',   
'MY\_min\_time', 'MY\_max\_time', 'MY\_mean\_time',   
'MZ\_min\_time', 'MZ\_max\_time', 'MZ\_mean\_time', 'rAlpha\_min\_time', 'rAlpha\_max\_time', 'rAlpha\_mean\_time', 'rBeta\_min\_time', 'rBeta\_max\_time', 'rBeta\_mean\_time', 'rGamma\_min\_time', 'rGamma\_max\_time', 'rGamma\_mean\_time', 'MGX\_min\_time', 'MGX\_max\_time', 'MGX\_mean\_time', 'MGY\_min\_time', 'MGY\_max\_time', 'MGY\_mean\_time', 'MGZ\_min\_time', 'MGZ\_max\_time', 'MGZ\_mean\_time',   
'key\_pressed', 'user\_id']

**Question** **3**: What is your code for extracting these simple time domain statistical features?  
(Total score: 20 points. Grading rubric:  
100% points for correct and complete code;  
60% points for partially correct and partially complete code;  
30% points for partially correct or partially complete code.)

### **LAB EXERCISE/STEP 3 (Extract Simple features in the Frequency domain)**

Besides the features in the time domain, statistical features in the frequency domain can also be useful. Please revise your code by enlarging the feature vector so that it includes statistical features from both the time domain and the frequency domain. In more details, you can use [Fast Fourier Transform](https://en.wikipedia.org/wiki/Fast_Fourier_transform) to convert each of the 9 time series to the frequency domain. Some of your features will be on the complex plane. You can use the Python *numpy* module for performing the Fast Fourier Transform. Your code will further take the minimum, maximum, and mean values of 9 types of frequency domain data and add 27 new frequency domain features. In other words, **now** each row is a record consisting of **56** fields as shown below, where the first **54** fields constitute the feature vector:

['MX\_min\_time', 'MX\_max\_time', 'MX\_mean\_time',  
'MX\_min\_freq, 'MX\_max\_freq, 'MX\_mean\_freq,  
'MY\_min\_time', 'MY\_max\_time', 'MY\_mean\_time',  
'MY\_min\_freq', 'MY\_max\_freq', 'MY\_mean\_freq',   
'MZ\_min\_time', 'MZ\_max\_time', 'MZ\_mean\_time',   
'MZ\_min\_freq', 'MZ\_max\_freq', 'MZ\_mean\_freq', 'rAlpha\_min\_time', 'rAlpha\_max\_time', 'rAlpha\_mean\_time',  
'rAlpha\_min\_freq', 'rAlpha\_max\_freq', 'rAlpha\_mean\_freq',   
'rBeta\_min\_time', 'rBeta\_max\_time', 'rBeta\_mean\_time',  
'rBeta\_min\_freq', 'rBeta\_max\_freq', 'rBeta\_mean\_freq',   
'rGamma\_min\_time', 'rGamma\_max\_time', 'rGamma\_mean\_time',  
'rGamma\_min\_freq', 'rGamma\_max\_freq', 'rGamma\_mean\_freq', 'MGX\_min\_time', 'MGX\_max\_time', 'MGX\_mean\_time',  
'MGX\_min\_freq', 'MGX\_max\_freq', 'MGX\_mean\_freq',  
'MGY\_min\_time', 'MGY\_max\_time', 'MGY\_mean\_time',   
'MGY\_min\_freq', 'MGY\_max\_freq', 'MGY\_mean\_freq', 'MGZ\_min\_time', 'MGZ\_max\_time', 'MGZ\_mean\_time',   
'MGZ\_min\_freq', 'MGZ\_max\_freq', 'MGZ\_mean\_freq',  
'key\_pressed', 'user\_id']

**Question 4**: Please briefly explain what frequency domain features are, and describe the relationship between frequency domain features and time domain features.  
(Total score: 10 points. Grading rubric:  
100% points for a clear explanation and a clear description;  
60% points for only a clear explanation or a clear description;  
30% points for a vague explanation and/or a vague description.)

**Question 5**: What is your code for performing the Fast Fourier Transform and for extracting these simple frequency domain statistical features?  
(Total score: 15 points. Grading rubric:  
100% points for correct and complete code;  
60% points for partially correct and partially complete code;  
30% points for partially correct or partially complete code.)

### **LAB EXERCISE/STEP 4 (Extract the Total energy features)**

In this step, you will add one more type of statistical features, the *total energy* features, to your feature vector. Here, the total energy is defined as the sum of squared values of a series. Please use the [energy calculation formula for a discrete-time signal](https://en.wikipedia.org/wiki/Energy_(signal_processing)) to derive the total energy features in both the time and frequency domains for each of the 9 types of data, thus adding 18 new total energy features. In other words, **now** each row is a record consisting of **74** fields as shown below, where the first **72** fields constitute the feature vector:

['MX\_min\_time', 'MX\_max\_time', 'MX\_mean\_time',  
'MX\_min\_freq, 'MX\_max\_freq, 'MX\_mean\_freq,  
'MX\_Energy\_time', 'MX\_Energy\_freq',  
'MY\_min\_time', 'MY\_max\_time', 'MY\_mean\_time',  
'MY\_min\_freq', 'MY\_max\_freq', 'MY\_mean\_freq',   
'MY\_Energy\_time', 'MY\_Energy\_freq',  
'MZ\_min\_time', 'MZ\_max\_time', 'MZ\_mean\_time',   
'MZ\_min\_freq', 'MZ\_max\_freq', 'MZ\_mean\_freq',  
'MZ\_Energy\_time', 'MZ\_Energy\_freq',  
'rAlpha\_min\_time', 'rAlpha\_max\_time', 'rAlpha\_mean\_time',  
'rAlpha\_min\_freq', 'rAlpha\_max\_freq', rAlpha\_mean\_freq',  
'rAlpha\_Energy\_time', 'rAlpha\_Energy\_freq',   
'rBeta\_min\_time', 'rBeta\_max\_time', 'rBeta\_mean\_time',  
'rBeta\_min\_freq', 'rBeta\_max\_freq', 'rBeta\_mean\_freq',  
'rBeta\_Energy\_time', 'rBeta\_Energy\_freq',  
'rGamma\_min\_time', 'rGamma\_max\_time', 'rGamma\_mean\_time',  
'rGamma\_min\_freq', 'rGamma\_max\_freq', 'rGamma\_mean\_freq',  
'rGamma\_Energy\_time', 'rGamma\_Energy\_freq',  
 'MGX\_min\_time', 'MGX\_max\_time', 'MGX\_mean\_time',  
'MGX\_min\_freq', 'MGX\_max\_freq', 'MGX\_mean\_freq',  
'MGX\_Energy\_time', 'MGX\_Energy\_freq',  
'MGY\_min\_time', 'MGY\_max\_time', 'MGY\_mean\_time',   
'MGY\_min\_freq', 'MGY\_max\_freq', 'MGY\_mean\_freq',  
'MGY\_Energy\_time', 'MGY\_Energy\_freq',  
'MGZ\_min\_time', 'MGZ\_max\_time', 'MGZ\_mean\_time',   
'MGZ\_min\_freq', 'MGZ\_max\_freq', 'MGZ\_mean\_freq',  
'MGZ\_Energy\_time', 'MGZ\_Energy\_freq',  
'key\_pressed', 'user\_id']

**Question 6**: Please explain why the total energy features could be useful.  
(Total score: 10 points. Grading rubric:  
100% points for a clear explanation;  
50% points for a vague explanation.)

**Question 7**: What is your code for extracting the total energy statistical features?  
(Total score: 15 points. Grading rubric:  
100% points for correct and complete code;  
60% points for partially correct and partially complete code;  
30% points for partially correct or partially complete code.)

### **LAB EXERCISE/STEP 5 (design and Extract additional features)**

In this step, you are encouraged to design and extract some additional features that may be useful for performing the input inference as well as user fingerprinting attacks. You may also consider to revise your current feature vector.

**Question 8**: Please describe your additional features or revisions with brief explanation.  
(Total score: 5 points. Grading rubric:  
100% points for a clear description and a clear explanation;  
60% points for only a clear description or a clear explanation;  
30% points for a vague description and/or a vague explanation.)

### **LAB EXERCISE/STEP 6 (generate the csv Feature file)**

Write code to save your entire matrix (for the features and labels) to a CSV (comma-separated values)featurefile named ***feature\_data.csv***, in which the first row contains the column headers of your feature vector and two labels, and each of the rest rows is a record with the corresponding 74 fields (if you did not further add or remove features after STEP 4).

**Question 9**: What is your code for generating your CSV feature file?  
(Total score: 5 points. Grading rubric:  
100% points for correct and complete code;  
60% points for partially correct and partially complete code;  
30% points for partially correct or partially complete code.)

### **Puzzler (N/A)**

This is an advanced activity for students who complete the regular activities early. N/A for this lab.

## What to submit

Please answer all the 9 questions in this lab exercise. Please feel free to directly reuse this Word document to provide and submit your answers. Please submit your final CSV feature file ***feature\_data.csv*** and your code that generates it as mentioned in STEP 6.