Finger Sense Classification

I. Objective:

Using acoustic-based features, we want to classify the type of objects which touch a surface. More specifically, we want to distinguish the type of finger inputs, including pad and knuckle based on their acoustic signatures. Finger touch information can be used to facilitate the interaction between users and touch and even non-touch devices.

II. Data:

The available training data set consists of approximately 20K instances. Each instance includes the recorded audio waveform with 256 samples at sampling frequency 48 KHz (almost 5 msec), time stamp and the type of finger input. The test data set on the other hand, has the same structure with approximately 10K instances with unknown finger input.

III. Features:

The ability to distinguish finger inputs relies on the physical principle that different materials generate different acoustic patterns and have different resonant frequencies. Therefore, features should be selected such that they reflect those patterns. Figure 1 shows typical Pad and knuckle waveforms.

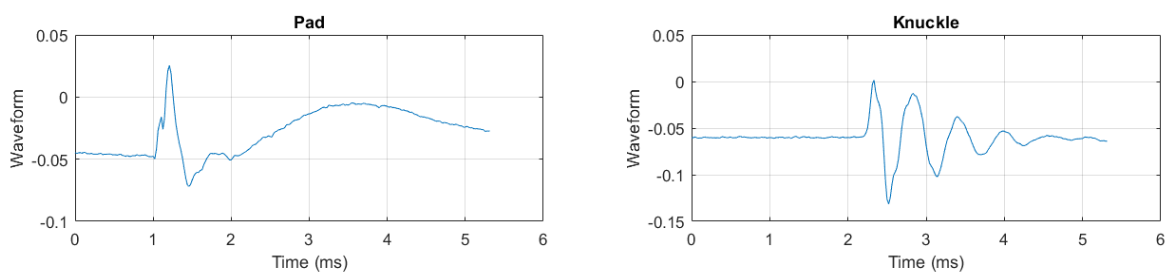

Figure 1: Pad and Knuckle audio waveforms

Based on previously reported successfully employed features, I use following features in the feature vector:

a) Frequency-domain features:

- First 500 FFT samples: A Fast Fourier Transform (FFT) of this window produces 2048 frequency bands. I consider e discard all but the lower 500 bands, representing the acoustic power from 0 to ~10kHz (feature indices 0-499)
- First 50 down-sampled FFT samples (feature indices 500-549)
- Average absolute amplitude of audio FFT (feature indices 550)
- Standard deviation of audio FFT (feature indices 552)
- Total absolute amplitude of audio FFT (feature indices 554)
- Center of mass of audio FFT (feature indices 556)

b) Time-domain features:
  - Average absolute amplitude of audio signal (feature indices 551)
  - Standard deviation of audio signal (feature indices 553)
  - Total absolute amplitude of audio signal (feature indices 555)
  - Center of mass of audio signal (feature indices 557)

c) Touch features:
  - (*x,y*) coordinate (feature indices 558-559)
  - Size of touch (feature indices 560-561)
  - Pressure (feature indices 562)
  - Orientation (feature indices 563)
  - Position (feature indices 564-565)

I use 6 frequency domain features, 4 time-domain features and 4 touch features for the classification purpose.

IV. Data preparation:

A. Data labels:

There are two classes in the training data set: *pad* and *knuckle.* I use these labels in the rest of this report and for the classification problem.

B. Categorical variables:

Most features in the database are real values, however the mobile device position (either hand or table) is a categorical variable. To avoid unfair penalty on this variable, I use dummy position variables. Dummy variable is the standard way of encoding categorical features using several binary features.

C. Training and test data:

To evaluate the performance of each classification technique, I split the data into two random subsets, train and test sets with %80 and %20 ratio, respectively. For the model cross-validation and optimization over model parameters, I use10-fold evaluation method.

D. **Variables:**

Almost all of the features in the database are real variables. However, they have different mean and variances. To avoid unfair penalty on these variables, I normalize all the variables.

The energy of the audio waveform is concentrated around the touch time and is zero for the rest of the time. If we add up all the audio waveforms, we get approximately a signal which is non-zero around the touch time and zero at the rest of the time.

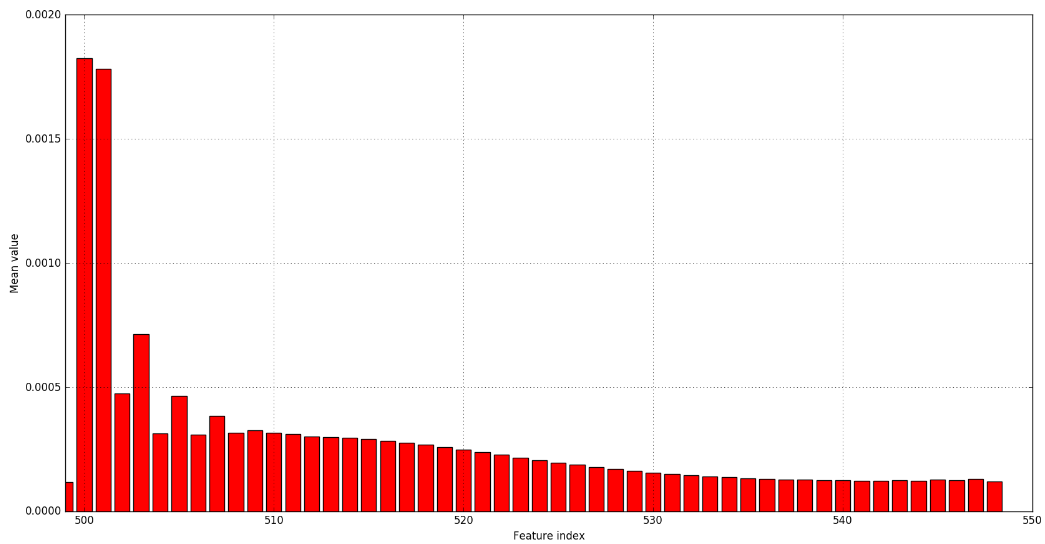

Figure 2: Mean value of the down-sampled audio FFTs which is the smoothed version of mean of the audio FFTs in Figure 3.

E. **Feature Importance:**

Not all of the features have the same importance. To get a better understanding of the nature of the classification problem, I use different feature selection models to understand the importance of each variable.

I use Random Forest model for feature ranking. Random Forest consists of a number of decision trees, where each node in the decision trees is a condition on a single feature, designed to split the dataset into two sets. The general idea is to permute the values of each feature and measure how much the permutation decreases the accuracy of the model. Clearly, for unimportant variables, the permutation should have little to no effect on the model accuracy, while permuting important variables should significantly decrease it. The feature importance is plotted in Figure 5.

As it can be seen from Figure 5, low frequency components (feature index <200 corresponding to frequencies less than 2 KHz) have little importance, while high frequency components play crucial role in the accuracy of the classification. This observation tallies the spectrogram analysis of pad and knuckle input types in Figure 2, where, both pad and knuckle spectrograms have approximately same level in low frequencies, but different levels for high frequencies.

The other important observation from Figure 5 is the periodic behavior of the feature importance measure, for feature indices 200 – 500 as they can be seen as harmonics. Specifically, this behavior is a result of Fourier transform where a frequency component also triggers all the harmonics.

This is worth to note that the frequency components in 2-3.5 KHz frequency band (feature indices 200-320) are important indicators for classification, however higher harmonics (3.5-10 KHz band corresponding to feature indices 320-500) are stronger indicators.

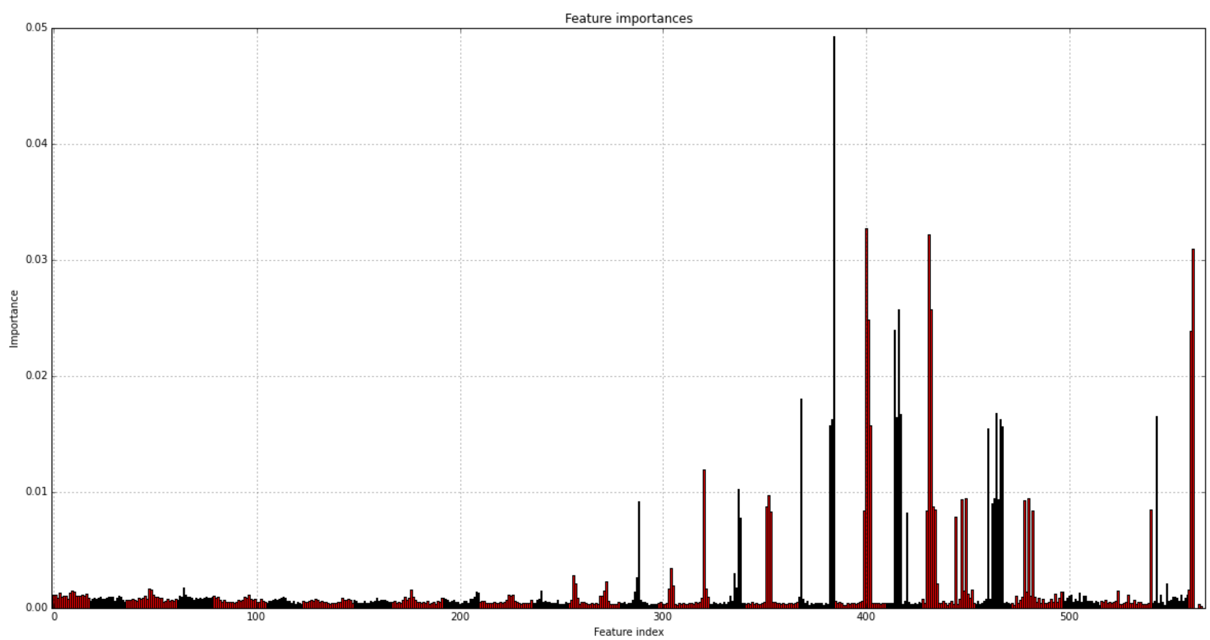

Figure 3: The importance of features in Random Forest methods where unimportant variables, have little contribution to the overall model accuracy.

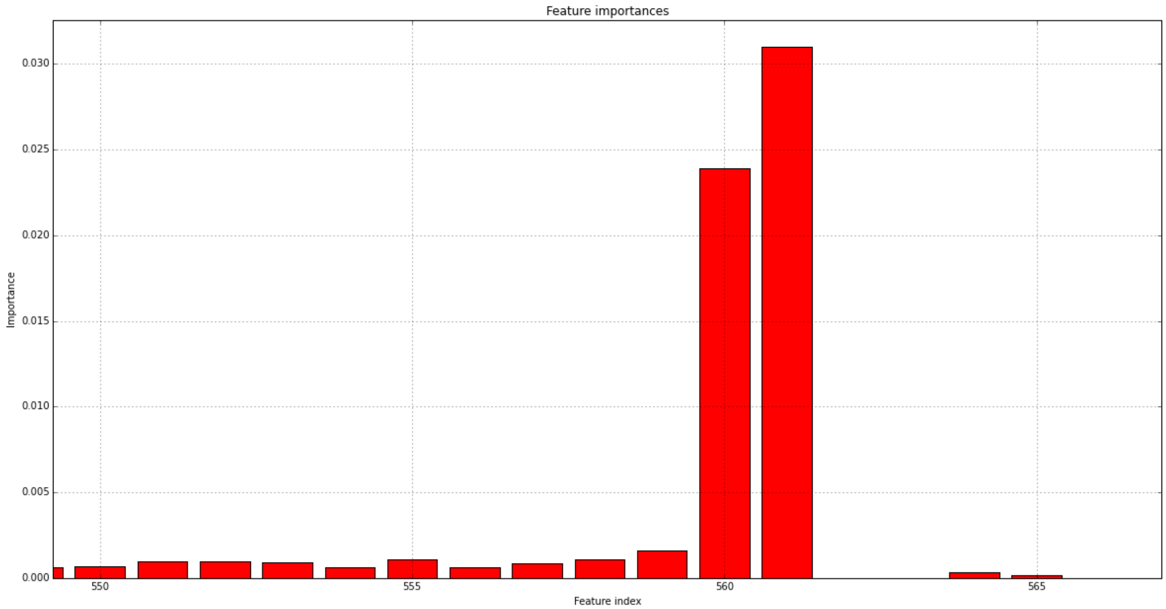

Figure 4: Signal aggregate features (feature indices 550-557) have no significance. Position information, feature indices 564-565 don't carry any meaningful information. However, size of touch indicators (major and minor, feature indices 560-561) are significant features.

From Figure 5, it can also be seen that signal aggregate features (feature indices 550-557) including average absolute amplitude, total absolute amplitude, standard deviation and center of mass parameters don't play important roles in the classification.

Figure 5 also shows that the only significant touch feature is the size of touch indicators (major and minor, feature indices 560-561).

V. **Modeling:**

Due to the classification nature of the problem, I consider three different common classification techniques, including Logistic Regression, Random Forest and Support Vector Machine. For parameters that are not directly learnt within classification model, I use grid search technique in the parameter space to optimize the performance score using 10-fold cross-validation.

F. **Logistic Regression:**

In the logistic regression (LR) model which is also known as Maximum Entropy, the feature weight vector is found by minimizing the negative log likelihood with an L2 regularization which is a convex, unconstrained and differentiable optimization problem. The optimization problem can be solved with any gradient based optimization technique. I tune the LR model over regularization parameters. LR achieves an accuracy of %93 over both training and test data sets.

Table 1: Performance metrics for Logistic Regression model for the test data set

	Precision	Recall	F1-score	Samples
Pad	0.93	0.94	0.93	8324
Knuckle	0.93	0.93	0.93	8203
Avg / Total	**0.93**	**0.93**	**0.93**	**16527**

In Table 1, the performance metrics including precision, recall and F1-score for the training set are denoted. Figure 6 shows the hyper-parameters for the LR model. As it can be seen, for the LR model, the FFT feature coefficients are also periodic.

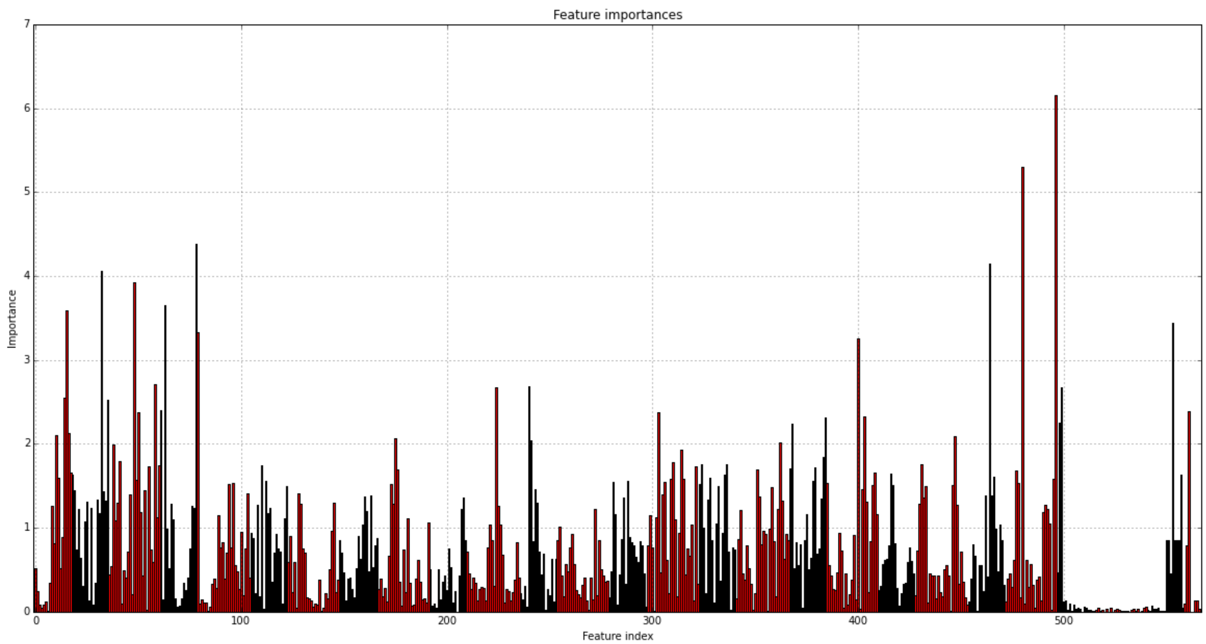

Figure 5: Hyper-parameters of the Logistic Regression model.

G. **Random Forest:**

Random forest (RF) model is another widely used classification method as a simple and effective solutions to supervised learning problems. RF model success extends to tasks across a variety of areas. In random forest, the predicted response for a (test) key is the vote of the prediction associated with the cell to which the key belongs among different trees. The tuning over the number of trees in the forest results to 70 trees in the forest. Random Forest model achieves %100 accuracy over the training set and %94 accuracy over the test data set.

Table 2: Performance metrics for Random Forest model over test set

	Precision	Recall	F1-score	Samples
Pad	0.93	0.95	0.94	2080
Knuckle	0.95	0.92	0.94	2052
Avg / Total	**0.94**	**0.94**	**0.94**	**4132**

H. **Support Vector Machine:**

Support Vector Machine (SVM) classification is another widely used classification method defined by a separating hyperplane. In addition to performing linear classification, using kernels, SVMs can efficiently perform a non-linear classification by implicitly mapping their inputs into high-dimensional feature spaces. Due to the computation complexity, I only proceed with linear kernel as I wan not able to tune the SVM classifer over kernels and regularization parameter. SVM can achiev %93 accuracy for both training and test data sets.

Table 3: Performance metrics for Support Vector Machin model over test set

	Precision	Recall	F1-score	Samples
Pad	0.92	0.94	0.93	2080
Knuckle	0.94	0.92	0.93	2052
Avg / Total	**0.93**	**0.93**	**0.93**	**4132**

VI. **Conclusion:**

Random Forest model has the highest accuracy, %94, over the test set. In addition, after the traing phase, it is fairly fast to be calculated for new samples in real time, which is independent of the size of the traing set. SVM is a close competitor, however, to get a better performance in the SVM model, we need to increase the number of traing samples. On the other hand, increasing the size of the traing set also increases the clculation time, making the SVM models slow for a real time implementation.

Logistic Regresion model also has a fixed calculation time for new samples, making it feasible for real-time applications. However, it uses non-zero coefficient even for a lot of highly corrolated features, making it difficult to have a clear interpretation and generalization.

Being fairly fast for both training and real-time implementation, having high accuracy, low dimensionality and a clear interpretation, I use the trained Random Forest model for the unlabled test data set. The classification results are presented in *fingersense-test-labels.csv* file.