Biometric Liveness and Anomaly Detection Project

Background

The idea of this project is to detect liveness of biometric input (such as brain signals) and prevent attacks. By liveness detection, we mean that input should have two properties: 1) belong to a live human being (not artificially generated), and 2) being captured at the current point in time (not being prerecorded). The project will be focused on ensuring the first properties of liveness. About the second feature, you will be writing about possible approaches and not necessarily implement them.

Introduction

You will be developing an app that acts as a verification server for liveness detection for brain signals. Data will be sent to app on the phone through some communication channel (Internet or Bluetooth), the app will analyze it (machine learning techniques) and announce the result (live or not live). The project will be focused on the first feature of liveness (live input that is not artificially generated). We will be providing datasets of brain signals for training and testing.

Instructions

Below you can find the workflow of the app and tasks needed to be completed.
They do not need to be done sequentially in the order given.

1) Create an app capable of receiving data through Internet or Bluetooth. I would suggest using Internet option since then you can easily test your system using any machine, and not constrained by having Bluetooth capability. For establishing internet communication, you can use a client server approach which standard libraries are available for it.

2) Raw brain signal is received as input by the app and needs to go through feature extraction.
    a. You might want to first do some preprocessing such as normalization (zero mean unit standard deviation), or frequency filtering.
    b. Try at least six diffirent feature extraction techniques. We will be providing a list of some possible features, which you can choose up to four features from it, and the rest of features will be of your choice.

3) Then the app should decide on liveness of the input.
    a. You will be having at least five trained machine learning models (check 4) on your system and will report the decision (live or fake) of each model on the phone, and afterwards do a voting on models, and report the final decision, and the true label of the input.
    b. The app will be having two modes; A) one single input is received and the operation is as described in 3.a, and B) a group of input is received, which then you will report the aggregate results (accuracy, false accept rate, false reject rate, half total error, F1 score, …) for each of the models, and for the voting model.

4) To train your models, you can do it either on a machine and move the trained model to your app, or you can use the mobile phone itself to train the model. In both cases, you will be reporting the time necessary for training.
    a. You will be having at least five models, which you can chooses up to 3 from the list we will provide, and the rest are of your choice.

5) For each model, you can decide between one-class classification or two-class classification. If you can utilize unsupervised learning or reinforcement learning techniques, there will be extra credit.

6) To test the model beyond the testing set provided by us, you will be creating new fake signals (attack vectors). There is no need to do this on the phone, and you can do it on a machine. There are three general approaches, but you are not limited to them. You need to create at least two new attack vectors, one should be using *signal generation* methods (check 6.b), and the other is of your choice.
   a. *Noise Addition:* You will add noise to original signals. Noise can be added in time domain, or in feature domain (e.g. frequency, wavelet, ..). If you decided to add noise in feature domain, you need to make sure the feature extraction method is reversible, so you can get back a new time domain signal.
   b. *Signal Generation*: One can train generative/predictive models to generate new signal. Any model used for time-series prediction/forecasting can also be used. Some models are ANFIS, GAN, VAE, ….
   c. *Random Inputs*: Create random signals in time domain, or random vectors in feature domain, and then map them back to time domain.

7) Provide execution time analysis for training, testing and attack generation.

8) Provide performance analysis (accuracy, false accept rate, false reject rate, half total error, F1 score, …) for training and testing of models.