Contents

Introduction

The objective of this session is to DEVELOP and EVALUATE an online signature recognition algorithm. According to the theory sessions, signature recognition systems can be divided into two categories:

- Off-line: the input is a static image of the signature.
- On-line: the signature is acquired using a specific digital sensor which includes the static image and dynamic signals related with the way the signature was done: x,y coordinates and pressure as a function of time.

Figure 1 shows a block diagram of a typical online signature recognition algorithm where [x,y,p] are the captured signals by the sensor (Cartesian coordinates and pressure), ft is the feature vector of the query signature to be compared with the fc feature vector of the signature stored in the database (claimed identity).
In this session we will assume that the data is available (previously acquired) and we will focus on the development of two modules:

- Feature Extraction Module.
- Matcher.

You must complete the tasks proposed in this document and answer the questions included.

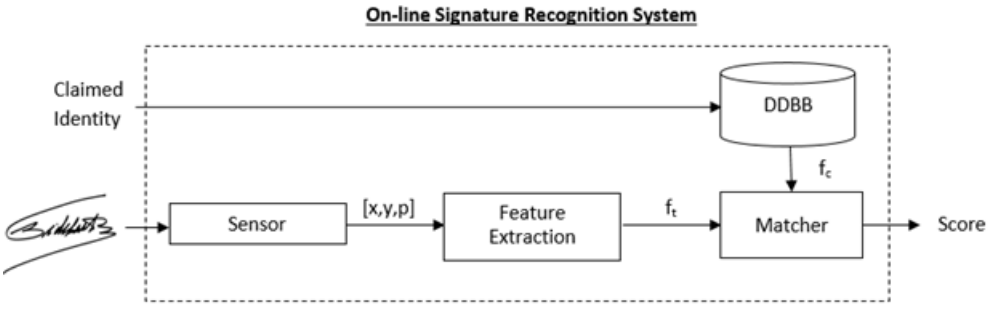

Figure 1: Block Diagram of a typical online signature recognition system

Data

For the practice we will use 50 users from the BiosecurID database. Each of the users have 28 signatures acquired in 4 sessions with a time lapse of 2 months. From the 28 signatures, 16 are genuine (4 per session) and 12 are forgers (3 per session). In this practice we will only consider the genuine signatures.
Each of the signatures is stored in .mat file which contains three vectors of same length with the x, y coordinates and the pressure as functions of time. The formatting of the files is `uXXXXsYYYY_sgZZZZ.mat`:

- XXXX: user number
- YYYY: session number
- ZZZZ: signature number

The GENUINE signatures of each session are those with $ZZZZ = [0001, 0002, 0006, 0007]$.The signatures with $ZZZZ = [0003, 0004, 0005]$ are the FORGERS and they will NOT be used in this practice.

Drawing signals

Choose a signature (from a random user) and show (assuming that the sensor has a 200 samples/second acquisition rate):

- Signal x as a function of signal y.
- Signal x as a function of time.
- Signal y as a function of time.
- Signal p as a function of time.

The first thing we do is to modularize the data reading to avoid very long scripts. Firstly, we realise that we can read the data easily iterating through three `for` loops and using a single line to read the file, by using `sprintf(\'%02d\',user)`, which converts the user number to a 2 decimal number. This way, we can avoid the `if` statement:

```
for user=1:n_users
  for session = (1:4)
    for sign_genuine = (1:4)
        %This is how to load the signatures:
        BiosecurID=load(['./DB/u100', sprintf('%02d',user) ,'s000', num2str(session),
                                '_sg000', num2str(genuine_signs(sign_genuine)), '.mat']);
```

The whole code to read the data is in the file `ReadData.m`

Repeat the process

Are the different signals reasonable? Do they have the same length? Why?

Feature Extraction

The comparison of signals with different lengths is not trivial. Therefore, we will extract 4 global parameters of each of the signatures. So, each signature will be represented by a feature vector with fixed size equal to 4. These parameters are:

1. Total duration of the signature: T
2. Number of pen-up (number of times the pen was lifted). It means the number of times (not the number of samples) that p is equal to 0.
3. Duration of pen-down (signal p is different to 0) Td divided by the total duration $T : Td/T$
4. Average pressure in pen-down (signal p is different to 0).

You have to develop 4 functions to extract each of the parameters:

1. `T=Ttotal(x)`
2. `Npu=Npenups(p)`
3. `Tpd=Tpendown(p)`
4. `Ppd=Ppendown(p)`

According to those functions, we will develop a new function with input data (x, y, p) of a given signature and output data the feature vector containing the 4 parameters (`FeatVect=featureExtractor(x,y,p)`). Based on your function featureExtractor you have to develop a program (`ProcessBiosecurID.m`) to extract all the feature vectors from the database and store it in a matrix with 3 dimensions:

- Dimension 1: number of user (1:50)
- Dimension 2: number of signature (1:16)
- Dimension 3: number of parameter (1:4)

You have to save this matrix into the file `BiosecurIDparameters.mat`.

Once you have the file `BiosecurIDparameters.mat`, you have to plot the distributions normalized between 0 and 1 (dividing by the total number of points of the distribution) for each of the 4 parameters.
HINT: You can use the Matlab functions hist.m and histc.m

QUESTION: Plot the 5 distributions.

Performance Evaluation

We will evaluate the performance of our system according to the number of signatures N in the enrollment set $(N = 1, N = 4 and N = 12)$.

The similarity score between a query/test signature and the enrollment signatures (signatures in the database) will be the Euclidean distance between feature vectors (vectors with 4 parameters). The final score will be the average score of the N comparisons (comparison between the query/test sample and the N enrollment samples).

You have to develop the function `Score=Matcher(test,Model)` where:

- Score: is the final score of the comparison.
- test: is the feature vector of the query/test signature (1×4)
- Model: is a matrix containing the feature vectors of the signatures enrolled in the database. Therefore, this matrix contains Nx4 values in which N is the number of signatures enrolled for the claimed identity.

There are two cases to be analyzed: Genuine Scores: scores obtained when you compare a signature with his real enrolled identity (claimed identity = enrolled identity). So these users should be accepted by the system. For each user you will use N signatures as enrolled samples and the rest for testing:

1. For $N = 1$ we will have $SG = 15$ genuine scores.
2. For $N = 4$ we will have $SG = 12$ genuine scores.
3. For $N = 12$ we will have SG=4 genuine scores.

For each of the scenarios $(N = 1, 4, 12)$ you have to save all the genuine scores into a matrix (with dimension $50 \times SG$). Each of the three matrixes will be stored into a .mat file with name: `GenuineScores_N.mat`.

Impostor Scores: scores obtained when you compare a signature with the enrolled samples of other users (claimed identity $\neq$ enrolled identity). So these users should be rejected by the system. In this case, we will compare one signature of each user (the first one) with the models of the rest of the users (excluding the genuine case). Therefore, we will obtain SI=49 impostor scores for each user and each scenario $(N = 1, 4, 12)$.

For each scenario $(N = 1, 4, 12)$ these impostor scores will be saved into a matrix with dimensions $50 \times SI$, $(50x49)$. Each of the three matrixes will be stored into a .mat file with name: `ImpostorScores_N.mat`.
Once we obtain the genuine and impostor scores, we will evaluate the performance of our system for each of the three scenarios $(N = 1, 4, 12)$ as a function of: FAR/FRR, EER and DET curves. To obtain these performance metrics you will have available the next functions:
`[EER]=Eval_Det(GenuineScores, ImpostorScores, 'b')`, where the parameters are

1. EER: value of the Equal Error Rate (error when FAR and FRR are equal)
2. `GenuineScores`: the scores from target or genuine comparisons. These scores are obtained after applying the following normalization: $GenuineScores = 1./(GenuineScores_N + 0.00000001)$
3. `ImpostorScores`: the scores form non target or impostor comparisons. These scores are obtained after applying the following normalization: $ImpostorScores = 1./(ImpostorScores_N + 0.00000001)$

QUESTION. Plot the performance graphics (DET curves) using the genuine and impostors score stored in their respectively matrixes (for each of the scenarios $N = 1, 4, 12$). Indicate the EER value.
QUESTION. According to the results, are they reasonable? What metrics are more illustrative? When do you obtain the best performance?

Extra

5.1 Extra work 1

If you want to obtain a mark up to 8 points out of 10 you should complete one of the following points:

1. Think and give a reasonable explanation of some additional features you can extract from the signatures. Program them, and repeat the point 4. Performance Evaluation in order to prove their improvement in the system performance.
2. Obtain a list of the most discriminative features and based on that make combinations of features in order to obtain a better performance.
3. Make an evaluation using the skilled forgeries signatures and compared the results with the random forgeries.

5.2 Extra work 2

Extra work 2: If you want to obtain a mark up to 10 points out of 10 you should do the following (directly, without doing the Extra work 1):

1. Develop an online signature recognition system based on local features (time functions) and Dynamic Time Warping for the Matcher. Repeat the same experimental protocol followed in the practice but using this new signature recognition system.
2. You should use the following local features (time functions): x, y, pressure, (and their corresponding first and second derivative).
3. You can use the DTW matcher available in Matlab. Take into account that it only allows to compare time functions of different lengths 1 to 1, i.e., x1 vs x2, y1 vs y2, etc. Therefore, you should compare time functions 1 to 1 and finally obtain the average between all time functions in order to obtain the final score of the comparison between two signatures.
4. The equation to obtain the score of the 1vs1 time function comparison is score=e-D/k, where D is the minimum accumulative distance obtain after using DTW in Matlab, and K is the number of aligned time samples.