MATLAB Implementation of [A keyless multimodal-based user authentication scheme using a generative adversarial network]

Overview

This repository contains the MATLAB implementation of the [A keyless multimodal-based user authentication scheme using a generative adversarial network]. The algorithms are designed to combine biometric features from multiple modalities, such as iris and face images, for a robust and secure identity verification process.

Prerequisites

Before running the code, ensure you have the following installed:

- **MATLAB (version R2024b or later)**

pseudo-Code

Algorithm 1 The Proposed Feature-Level Fusion Scheme

Input: A set of p training left iris binary samples each of length $n\{L_1,..., L_p\}$, right iris binary samples each of length $n\{R_1,..., R_p\}$, face binary samples each of length $f\{F_1,..., F_p\}$, N *is* the number of training epochs for the GAN model, *alpha* is the learning rate for the GAN model, h_1 is the generator network's hidden layer size, and h_2 the discriminator network's hidden layer size.

1: Create x_i a template of size $(2n+f)$ by concatenating L_i, R_i, and F_i for each sample i: $i \in \{1,..., p\}$.
2: Randomly select a template x_i: $i \in \{1,..., p\}$.
3: Create a salting key k, a permutated version of x_i.
4: Create a reference template T for each enrolled user using equation (9).
5: Create a binarized template Y for each enrolled user using equation (10).
6: Construct a fully connected multilayer generator network of size $[(2n+f) \times h_1 \times (2n+f)]$.
7: Construct a fully connected multilayer discriminator network of size $[(2n+f) \times h_2 \times 1]$.
8: Randomly initialize the generator network's weight matrix, W_g
9: Randomly initialize the discriminator network's weight matrix, W_d
10: Train the generator and the discriminator network alternatively using the back-prorogation algorithm using $\{x_1,..., x_p\}$ as inputs for the generator network and k as input to the discriminator network.
11: After N epochs of training, retrieve the generator network's output, S.
12: Create the reference cancelable template for the enrolled user, C_{ref}, using equation (8).
13: Store W_g and C_{ref} in the system's database for the authentication phase.

Output: In the authentication phase, the decision is made for the test cancelable template C_{test} using equation (11).

Algorithm 2 The Proposed GAN-based/ Decision-Level Fusion Schemes

Input: A set of p training left iris binary samples each of length n {L_1,..., L_p }, right iris binary samples each of length n { R_1 ,..., R_p }, face binary samples each of length f{ F_1 ,..., F_p }, N *is* the number of training epochs for the GAN models, *alpha* is the learning rate, h_1 is the generator networks' hidden layer size, and $h_2$is the discriminator networks' hidden layer size.

1: Randomly select left iris template l_i: $i \in \{1,..., p\}$.
2: Randomly select right iris template r_i:$i \in \{1,..., p\}$.
3: Randomly select face template f_i: I $\in \{1,..., p\}$.
4: Create salting keys k_l, k_r, and k_f, permutated versions of l_i,r_i, *and* f_i, *respectively.*
5: Create reference templates T_l, T_r,and T_f for each enrolled user using equation (9).
6: Create binarized templates Y_l, *Yr, and* Y_f for each enrolled user using equation (10).
7: Construct three independent generator networks of sizes [$n \times h_1 \times n$], [$n \times h_1 \times n$], and [$f \times h_1 \times f$].
8: Construct three independent discriminator networks of sizes [$n \times h_2 \times 1$], [$n \times h_2 \times 1$], and [$f \times h_2 \times 1$].
9: Randomly initialize generator networks' weight matrices.
10: Randomly initialize discriminator networks' weight matrices.
11: Train the generator and the discriminator networks for each model using the back prorogation algorithm using {L_1,..., L_p },{ R_1,..., R_p }, and { F_1,..., F_p }as inputs for each generator network, respectively, and k_l, k_r, and k_f as inputs to each discriminator network, respectively.
12: After N epochs of training, retrieve generator networks' output, S_l, S_r, and S_f.
13: Create reference cancelable templates for each model, C_{l_ref}, C_{r_ref}, and C_{f_ref} using equation (8).
14: For the GAN-based fusion scheme, create C_{ref}, a size *(2n+f)* template by concatenating C_{l_ref}, C_{r_ref}, and C_{f_ref}.
15: Store generator networks' weight matrices and reference template(s)in the authentication system's database

Output: The authentication phase for GAN-based is made for the test cancelable template C_{test} using equation (11). Authentication phase for the Decision-Level scheme is made using majority voting for the test cancelable templates C_{l_test} , C_{r_test} , and C_{f_test} using equation (5).

Installation

1. **Clone the repository:**

```
git clone https://www.mediafire.com/folder/l4mk86dr0o295/%20Implementation
```

2. **Navigate to the project directory.**

3. **Open MATLAB and set the project directory as the working directory.**

Usage

To run the algorithm, execute the following command in the MATLAB Command Window:

1. **For running the Feature-Level Fusion Scheme:**
   `Multi_biometric_Bio_GAN_Feature_Fusion.m`
2. **For running the GAN-Based Fusion Scheme:**
   `Multi_biometrics_Bio_GAN_based_fusion.m`
3. **For running the Decision -Level Fusion Scheme:**
   `Multi_biometrics_Bio_GAN_Decision_fusion.m`

Input

- **The process starts by loading biometric data from a specified directory, which includes face, left iris, and right iris images for each user.**

Output

- **Results are saved for intra-class and inter-class, tests are saved in a .mat file for later analysis** (`save(path, 'hd_intra', 'hd_inter')`)
- **The Performance metrices are visualized, includes [**EER value**,** intra-class and inter-class distributions of the Hamming distances, ROC curve and plots FAR and FRR**].**