S.

**Ain Shams University**

**Faculty of Computer & Information Sciences**

**Scientific Computing Department**

**Handwritten Signature Identification and Verification**

**By**

**SC\_34**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **No** |
| 20191700047 | أحمد عبدالجواد رمضان عبدالجواد | 1 |
| 20191700225 | خالد شريف عبداللطيف عبد المجيد عزام | 2 |
| 20191700284 | سعد وليد سعد على ابوحسن | 3 |
| 20191700785 | يوسف محمد جمعه محمد | 4 |
| 20191700138 | امير ممدوح حلمى شاكر | 5 |
| 20191700397 | عمر أحمد الشناوى عبدالحميد | 6 |

**Data preparation**

1. Read image paths using the **OS** library while ignoring **CSV** files at stage 1.
2. Passed the paths to OpenCV to read images and resize them based on the technique we are going to use.
   * **Stage 1**
     1. **HOG** model image sizes were (64, 128, 1)
     2. **CNN** model image sizes were (128, 128, 1)

* **Stage 2**
  1. **BOW** model image sizes were (128, 128, 3)
  2. **Siamese** model image sizes were (128, 128, 3)

1. Created labels for CNN model at stage 1, labels were one hot encoded (e.g [0, 0, 0, 1, 0] for personD ), labels for the **HOG** model were a number between 0 and 4, the same technique were used in **BOW**, but classes were either 0 or 1.
2. Same methods were used in Stage 3 but additionally text files were read corresponding to each image in the data set.

**Models and Techniques**

1. **Signature Identification (Stage 1)**

For the First Stage, The task was to apply image classification on signatures to **Identify** who it belongs to. We used two different models for that task.

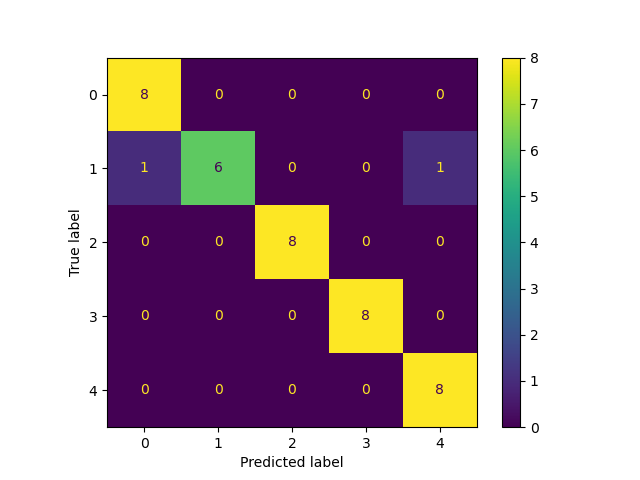
* 1. **CNN – Model**

We applied simple **CNN** architecture consisting of 4 convolution layers and 1 pooling layer, followed by 2 fully connected layers.

**The full Architecture is:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Layer** | **# filters/ neurons** | **Filter Size** | **Stride** | **Size of feature map** | **Activation Function** |
| Input | - | - | - | 128 x 128 x 1 | - |
| Conv 1 | 32 | 5x5 | 2 | 62 x 62 x 32 | ReLU |
| Conv 2 | 32 | 5x5 | 1 | 58 x 58 x 32 | ReLU |
| Max Pool 1 | - | 3x3 | 1 | 19 x 19 x 32 | - |
| Conv 3 | 32 | 5x5 | 2 | 8 x 8 x 32 | ReLU |
| Conv 4 | 32 | 5x5 | 1 | 4 x 4 x 32 | ReLU |
| FC 1 | - | - | - | 512 | ReLU |
| FC 2 | - | - | - | 128 | ReLU |
| FC 3 | - | - | - | 5 | Softmax |

**Performance Measures**



|  |  |  |
| --- | --- | --- |
| **-** | **Accuracy** | **Time** |
| **Training** | 0.985 | 51.47 s |
| **Testing** | 0.95 | 22.19 s |

* 1. **HOG – Model**

We applied **Histogram of Oriented Gradients** to extract features from images and then used these features to feed **Logistic Regression** model to classify them.

**HOG** basically is a feature descriptor that extracts the most important information about the image, specifically the edges and its direction, these are extracted by calculating the gradient in the **X** and **Y** directions then calculating the magnitude to find it is orientation, then it builds a histogram that shows the distribution of the data.

* + 1. **Performance Measures**

|  |  |  |
| --- | --- | --- |
| **-** | **Accuracy** | **Time** |
| **Training** | 1.0 | 2.04 s |
| **Testing** | 0.975 | 2.47 s |

* + 1. **Confusion Matrix**

A picture containing shape

Description automatically generated

1. **Signature Verification (Stage 2)**

For the Second Stage, The task was to apply image classification on the signatures of each person to **Verify** whether it is forged or real. We used two different models for that task.

* 1. **BOW – Model**

We applied 5 **Bag of Word** models, a model for each person. To extract features from images then feed these features to **Logistic Regression** model to classify them.

**Bag of Visual Words** basically is a technique to describe images and compute the similarity between them. At first, we use feature descriptors like **SIFT** on all our images to extract a list of descriptors of our dataset. Then we stack these descriptors and feed that stack to **K-mean** model to group these descriptors into **K** visual words. Then we build a histogram for these visual words by checking the appearance of the visual word in each image and our dictionary. Finally, this dictionary is fed to **Logistic Regression** model to classify them.