2023
2022

Handwritten Signature Identification and Verification

Marwan Mohamed Abd El-Halim El-Sayed	20191700620
Sara Adel El-Gebaly Mouhamed	20191700274
Mahmoud Mohamed Ahmed Orman	20191700606
Kirolos Nabil Mounir Fahmy	20191700460
Youssef Nader Michel Sobhy	20191700793

Data Preparation:

Classification Model:

Using **Image Preprocessing** class

- Normalize a picture pixel to 0-1 float (instead of 0-255 int).
- Add sample wise zero center (Zero center each sample by subtracting it by its mean).
- Add feature wise stdnorm (Scale each sample by the specified standard deviation. If no specified, std is evaluated over all samples data.

Using **Image Augmentation** class

This class is meant to be used as an argument of `input_data`. When training a model, the defined augmentation methods will be applied at training time only. Note that Image Preprocessing is like Image Augmentation but applies at both training time and testing time.

- Add random flip left right
- Add random rotation Randomly rotate an image by a random angle (-max_angle, max_angle).

Siamese Model:

- We used Pretrained weights of **ImageNet**
- Used batch normalization in the non-trainable layer
- Use **batch generator** function as preparation function to reduce the complexity of the memory

Models' descriptions and techniques:

Classification using Convolutional neural network model:

- Use **'tflearn'** modules to train the model to classify between different persons' signatures
- Use **One-hot encoding** technique to label our training data
    - Person A -> [1,0,0,0,0]
    - Person B -> [0,1,0,0,0]
    - Person c -> [0,0,1,0,0]
    - Person D -> [0,0,0,1,0]
    - Person E -> [0,0,0,01]
- Use convnet_cifar10 CNN architecture
- Use **softmax** as activation function in output layer with 5 neurons.

Classification using Siamese model:

- Use **Siamese** techniques which use Euclidean distance between two feature vectors .
- Use triplet loss concept to train the distance function.
- $l = \max\left(d(a,p) - d(a,n) + margin\,, 0\right)$, where a is anchor, p is positive, and n is negative sample.

Object detection model using faster R-CNN:

- We take an image as input and pass it to the **ConvNet** which returns the feature map for that image.
- **Region proposal network** is applied on these feature maps. This returns the object proposals along with their object-ness score.
- A **ROI** pooling layer is applied on these proposals to bring down all the proposals to the same size
- Finally, the proposals are passed to a **fully connected layer** which has a SoftMax layer and a linear regression layer at its top, **to classify and output the bounding boxes for objects.**

Time:

Handwritten Signature Identification:

- Train -> 10 min (avg 44 sec for each epoch).
- Test -> 2 seconds

Handwritten Signature Verification:

- Train -> 955 seconds (avg 60 sec per epoch).

```
EPOCH: 13    (Epoch done in 60 sec)
Loss on train    = 0.00000
1/1 [==============================] - 1s 975ms/step
Accuracy on test = 0.93333

EPOCH: 14    (Epoch done in 60 sec)
Loss on train    = 0.00000
1/1 [==============================] - 1s 971ms/step
Accuracy on test = 0.93333

EPOCH: 15    (Epoch done in 59 sec)
Loss on train    = 0.00000
1/1 [==============================] - 1s 986ms/step
Accuracy on test = 0.93333
Train Time: 955.3327951431274 Seconds
Model: "Encode_Model"
```

Image Classification accuracy:

Train Accuracy:

- Train data -> 160
- Validation data -> 40

```
Training Step: 29  | total loss: 0.20406 | time: 27.644s
| Adam | epoch: 010 | loss: 0.20406 - acc: 0.9121 -- iter: 128/160
Training Step: 30  | total loss: 0.21767 | time: 44.200s
| Adam | epoch: 010 | loss: 0.21767 - acc: 0.9181 | val_loss: 0.06057 - val_acc: 1.0000 -- iter: 160/160
--
```

Figure 1: Classification Train and validation accuracy

Train accuracy = 91.81%

Validation accuracy = 100.0%

Test Accuracy:

- Test data -> 40

```
[4, 0, 4, 0, 1, 4, 3, 4, 2, 2, 3, 2, 3, 2, 1, 4, 1, 1, 4, 2, 0, 2, 0, 1, 0, 2, 4, 3, 4, 3, 3, 3, 0, 1, 2, 0, 0, 3, 1, 1]
[4, 0, 4, 0, 1, 4, 3, 4, 2, 2, 3, 2, 3, 2, 1, 4, 1, 1, 4, 2, 0, 2, 0, 1, 0, 2, 4, 3, 4, 3, 3, 3, 0, 1, 2, 0, 0, 3, 1, 1]
Test acc = 100.0%
```

Figure 2:classification test accuracy

Test accuracy = 100.0%

Siamese Classification accuracy:

Train accuracy:

Test accuracy:

```
EPOCH: 13    (Epoch done in 60 sec)
Loss on train    = 0.00000
1/1 [==============================] - 1s 975ms/step
Accuracy on test = 0.93333

EPOCH: 14    (Epoch done in 60 sec)
Loss on train    = 0.00000
1/1 [==============================] - 1s 971ms/step
Accuracy on test = 0.93333

EPOCH: 15    (Epoch done in 59 sec)
Loss on train    = 0.00000
1/1 [==============================] - 1s 986ms/step
Accuracy on test = 0.93333
Train Time: 955.3327951431274 Seconds
Model: "Encode_Model"
```

Figure 3:Siamese Train

Test accuracy = 93.3%