Team 26 :

107065436 莫尼巴 (MATE Teaka)
X1085032 VENANT-VALERY Thomas
X1085033 TIREL Nicolas

Introduction to Intelligent Computing : Final Project Report

Methodology

The first thing we did about our classifier was to fix the labels after having imported the data. So we replaced spaces by "_". The model we choose for our classifier is a CNN classifier. The different parameters we used are : 30 classes, a width and height of 64, 8000 images per class, 800 valid images per class and 1 test image per class. The next step was to code a drawing function. We also used another database to have more data so we can train more our model. We reshaped, normalized and split the data into X_train, Y_train and validation data set. Then, we tried to improve our model using a more complex one.

As for the generator, we used bot DCGAN and ACGAN but the methodology between them is almost the same. We import the data like we did for the classifier and normalize the images. Then, we defined the generator model and also the discriminator model. The discriminator evaluates the drawings giving them a score : a positive one is a "good" image and a negative one a "bad" image. We added checkpoints in order to be able to backtrack if needed.

How to train the model

The classifier uses a categorical cross entropy function to evaluate the loss and the Adam optimizer. We use two callbacks : an early stopping function and the reduceLROnPlat one. We use a batch size of 32, 50 epochs and a validation split of 5% (***Illustration 1***). We printed two graphs measuring the accuracy and the loss based on the epochs number. Those results were decent but they can be improved (***Illustration 2***), so we tried to improve our model. It has more layers and we only use 20 epochs this time (***Illustration 3***). We printed the graphs once more to notice that it improved our results (***Illustration 4***). We tried to improve it once more using data augmentation and then training it on 100 epochs.

We trained the generator using 100 epochs and a DC/ACGAN (***Illustration 6***). We also used a loss measurement function and the Adam optimizer. Then, we printed some of the drawings so we were able to see how accurate our model is. Our discriminator function also helped to remove automatically not significant drawings (***Illustration 5***).

Test Result

The test results of our classifier evolved since we improved our model twice. First, the testing accuracy was around 38% (***Illustration 2***). But with our next improvement, our score improved a lot to the value of approximately 70% (***Illustration 4***). However, the last model improvement did not change much our previous results.

To measure the test results of the generator, it was more subjective because of the fact that only us can tell if the drawing is accurate or not, but with the discriminator function, we ended up having a really good precision among test drawings (***Illustrations 7&8***).

Demo Result

???

Discussion

This project was a first approach of AI recognition and generation for us so we all learned a lot doing it. We met some issues doing it such as the GPU limitation. None of us had one, so we used GPU mode on Google Collab. However, since the training and prediction processes can be very long, we sometimes got disconnected so we have to start over again. In addition, our misunderstanding of the demo part may decrease our final grade a lot which disappoints us since we were confident about our models.

Conclusion

The main goal of this project was to be able to create a model that can classify but mainly generate accurate drawings. Using a CNN classifier and GANs, we think that we managed to develop models that will meet this project expectations.
As a last word, we can say that since the AI recognition and generation domains are expanding a lot nowadays, this project gave us some tools to know more about them. Thus, it may help us for the following of our studies and eventually our future jobs.

Illustrations

```
Model: "sequential_8"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_32 (Conv2D)           (None, 64, 64, 32)        320
_________________________________________________________________
max_pooling2d_20 (MaxPooling (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_33 (Conv2D)           (None, 32, 32, 64)        18496
_________________________________________________________________
max_pooling2d_21 (MaxPooling (None, 16, 16, 64)        0
_________________________________________________________________
dropout_12 (Dropout)         (None, 16, 16, 64)        0
_________________________________________________________________
flatten_8 (Flatten)          (None, 16384)             0
_________________________________________________________________
dense_16 (Dense)             (None, 680)               11141800
_________________________________________________________________
dropout_13 (Dropout)         (None, 680)               0
_________________________________________________________________
dense_17 (Dense)             (None, 30)                20430
=================================================================
Total params: 11,181,046
Trainable params: 11,181,046
Non-trainable params: 0
```

Illustration 1: 1st classifier model

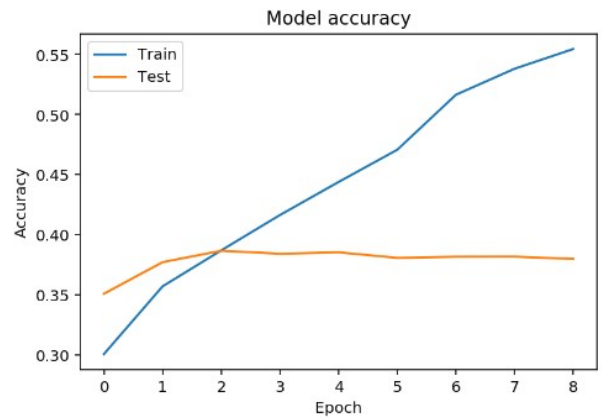

Illustration 2: 1st classifier results

```
Model: "sequential_10"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_40 (Conv2D)           (None, 64, 64, 16)        144
_________________________________________________________________
batch_normalization_18 (Batc (None, 64, 64, 16)        64
_________________________________________________________________
activation_18 (Activation)   (None, 64, 64, 16)        0
_________________________________________________________________
conv2d_41 (Conv2D)           (None, 64, 64, 16)        2304
_________________________________________________________________
batch_normalization_19 (Batc (None, 64, 64, 16)        64
_________________________________________________________________
activation_19 (Activation)   (None, 64, 64, 16)        0
_________________________________________________________________
max_pooling2d_25 (MaxPooling (None, 32, 32, 16)        0
_________________________________________________________________
conv2d_42 (Conv2D)           (None, 32, 32, 32)        4608
_________________________________________________________________
batch_normalization_20 (Batc (None, 32, 32, 32)        128
_________________________________________________________________
activation_20 (Activation)   (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_43 (Conv2D)           (None, 32, 32, 32)        9216
_________________________________________________________________
batch_normalization_21 (Batc (None, 32, 32, 32)        128
_________________________________________________________________
activation_21 (Activation)   (None, 32, 32, 32)        0
_________________________________________________________________
max_pooling2d_26 (MaxPooling (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_44 (Conv2D)           (None, 16, 16, 64)        18432
_________________________________________________________________
batch_normalization_22 (Batc (None, 16, 16, 64)        256
_________________________________________________________________
activation_22 (Activation)   (None, 16, 16, 64)        0
_________________________________________________________________
conv2d_45 (Conv2D)           (None, 16, 16, 64)        36864
_________________________________________________________________
batch_normalization_23 (Batc (None, 16, 16, 64)        256
_________________________________________________________________
activation_23 (Activation)   (None, 16, 16, 64)        0
_________________________________________________________________
max_pooling2d_27 (MaxPooling (None, 8, 8, 64)          0
_________________________________________________________________
dropout_15 (Dropout)         (None, 8, 8, 64)          0
_________________________________________________________________
flatten_10 (Flatten)         (None, 4096)              0
_________________________________________________________________
dense_20 (Dense)             (None, 512)               2097664
_________________________________________________________________
dense_21 (Dense)             (None, 30)                15390
=================================================================
```

Illustration 3: 2nd classifier model

```
Discriminator model:
Model: "model_9"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_16 (InputLayer)           [(None, 28, 28, 1)]  0
__________________________________________________________________________________________________
sequential_6 (Sequential)       (None, 12544)        387840      input_16[0][0]
__________________________________________________________________________________________________
generation (Dense)              (None, 1)            12545       sequential_6[1][0]
__________________________________________________________________________________________________
auxiliary (Dense)               (None, 30)           376350      sequential_6[1][0]
==================================================================================================
Total params: 776,735
Trainable params: 776,735
Non-trainable params: 0
__________________________________________________________________________________________________
```

Illustration 5: Discriminator model

```
Combined model:
Model: "model_11"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_19 (InputLayer)           [(None, 100)]        0
__________________________________________________________________________________________________
input_20 (InputLayer)           [(None, 1)]          0
__________________________________________________________________________________________________
model_10 (Model)                (None, 28, 28, 1)    2659897     input_19[0][0]
                                                                 input_20[0][0]
__________________________________________________________________________________________________
model_9 (Model)                 [(None, 1), (None, 3 776735      model_10[1][0]
==================================================================================================
Total params: 3,436,632
Trainable params: 2,659,321
Non-trainable params: 777,311
__________________________________________________________________________________________________
```

Illustration 6: Generator model

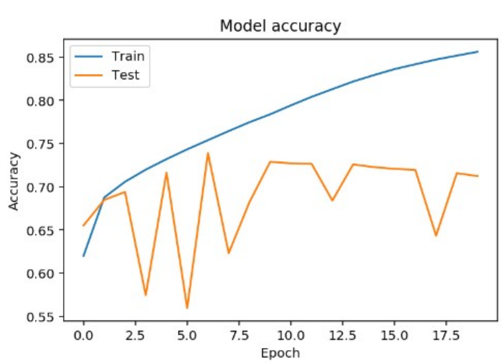

Illustration 4: 2nd classifier results

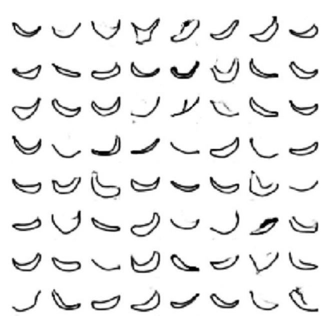

Illustration 7: 64 generated bananas

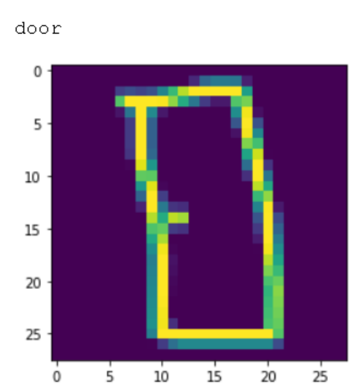

Illustration 8: Generated drawing and its label