CS401 : Eyetracking Project

Jace Egenbacher International Undergraduate
Pierre Kovarski Erasmus Undergraduate

Repartition of the work

Pierre Kovarski- Data Treatment, Results analysis
Jace Egenbacher- ML Model

Libraries Needed

Pandas
Tensorflow
Keras
Sklearn
Matplotlib

Data treatment

As advised by the report made by the searchers, 3 files only have been used to build this model :

- rawBlinkLogicalL.csv : blink of the left eye
- microsaccadeBinoc.csv : microsaccades
- interpPupilL.csv : pupil diameter of the left eye

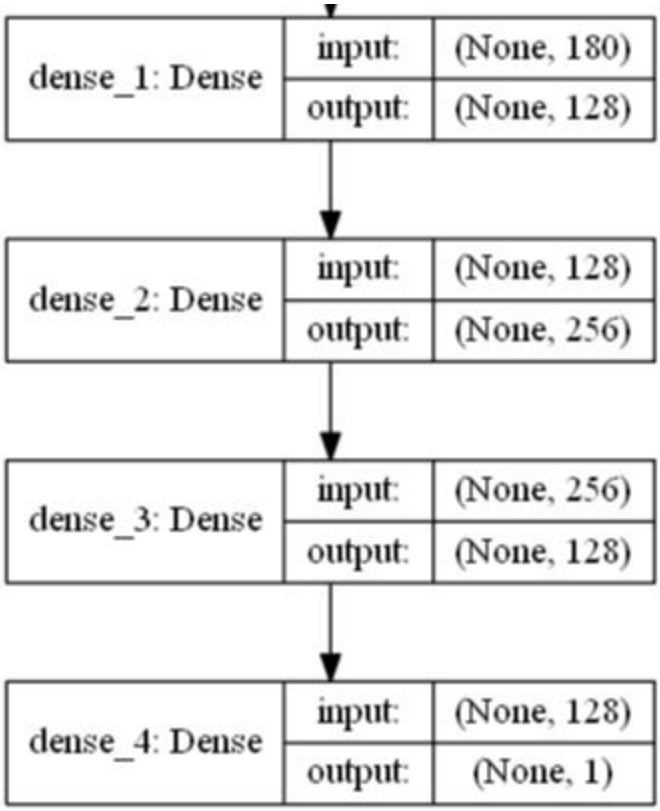
Only the values between 6s and 9s have been kept and we decided to average every 50 points because it is what provided the best results.
We replaced also 1 : CP (perceived) and 0 : CNP (non perceived).
The first issue we faced is that is the dataset is unbalanced : 3700 samples are recognized as perceived while 2122 are recognized as non perceived.

At the first, our neural network just predicted only predicted 1 so and got 63% right, so I create a training set composed of half 0 and half 1.

However because of this, we haven't performed cross-validation, we just shuffle the dataset and relaunch the script.We have a training set equal at 90% of the size of the set and a testing set equal at 10% of the size.

Method

Used Keras on top of Tensorflow to create a MLP model with 4 dense layers, 3 using a hyperbolic tangent activation function and the 4th using a sigmoid squashing function. The model is utilizing the Nesterov Adam optimizer with default values as recommended.



Results

The neural network model gives us an array of probabilities such as :

```
>>> prediction_prob
array([[0.3133052 ],
       [0.41286138],
       [0.57491255],
       ...,
       [0.3476665 ],
       [0.42460722],
       [0.61318827]], dtype=float32)
```

In order to have a precision higher to 0.9, we compute the number of true positives (tp) and false positives (fp), every 0.01 from 0.45 to 0.6 in the probability array and we stop when we have tp > 9*fp.

The output is like :

```
false positive : 22
>>> exec(open("./main.py").read())
[[687  21]
 [741 461]]
precision : 0.956432
 true positives : 461
 false positive : 21
```

The first line corresponds to a matrix which is the confusion matrix such as this one

False Negative	False Positive
True Negative	True Positive

After several runs instead of cross-validation (we just shuffle every time), predicted results of this model include:

- Precision > .90
- 400 < TP+FP < 600
- Accuracy ~ .75