# capstone project

# Machine Learning Engineer Nanodegree

## Capstone Project

Marcial Galván, PhD
July 7th, 2019

## I. Definition

### Project Overview

Early and objective detection of people with Autism Spectrum Disorder (ASD) is a challenge in actual neuroscience, and it is important in order to offer the right treatment to the patients at early ages.

The diagnosis of people with ASD is made by the use of behavioral exams and questionnaires that require considerable time and expertise on the side clinicians and parents, so is biased by the evaluator. There are some standardized tests, as the Autism Spectrum Quotient (AQ test) [S. Baron Cohen, Wheelwright, Skinner, Martin, & Clubley, 2001], that allows a classification of the autism degree, but it has limitations in the application. Otherwise, those test are easily falsifiable by the subjects, and can´t be applied to infants.

In order to obtain an objective detection of autism, my research group, Neurochemistry and Neuroimaging Laboratory, University of La Laguna, is trying two ways. The first one is biological: the use of Functional Magnetic Resonance Spectroscopy of the brain (fMRS), in the search of metabolite markers. The second is the use of the gaze exploration patterns as indicator of the autism presence: People with autism has a different relation with the world, and it has a reflection in how they explore images.

This second approach is the one we use in this project.

At the time of the capstone proposal we didn´t knew if this problem has a solution, now, thanks to this work, we think is a promising path of research.

### Problem Statement

There is no clear inner-separation between people with ASD (it is an spectrum of features). Also it is presumed that there is a part of the population of people classified a typically developing (TD) that have features belonging to ASD and have not been diagnosed, so we cannot assure that our TD patients are absolutely ASD features free.
The Brain with autism has a different behavior than a normal one, their degrees of attention and objects of interest, and it is reflected in how people with autism sees and explores the world.

Based on that fact we will try to use the gaze exploratory behavior as biomarker of the existence of autism.
We had used a dataset obtained in an experimental setup with consist in gaze data obtained by an eye tracker while experimental subjects are seeing a series of pictures.

In the data collection subject are exposed to a sequence of 22 different images, separated by the image of a cross. The subject is requested to center the look on the cross when it appears, acting as starting point of view for freely explore the images that came after it. Their gazing point on the screen, over all the sequence, is recorded as a temporal sequence. These gaze travelling are the pieces of our dataset, for each subject we collect 22 different image exploration patterns and 23 cross search patterns.

Our initial guess was to use the gaze data from image exploration to obtain characteristics of the exploratory behavior, but, as we will see in the data exploration analysis, we had to change the approach due to the low quality of the data (it has helped us to redesign the experimental protocol).

To summarize, the complete problem process we are looking for is:

We look for an automatic classification of people as belonging to two groups, ASD (Autism Spectrum Disorder) or Typically developing (TD), for that:

1. A subject is exposed to a series of images
2. The gaze behavior of the subject is recorded by means of an eye tracker.
3. The recorded gaze is processed and feed our prediction model.
4. The model tells us if the subject is a person with ASD or not.

**Metrics**

Our target is to detect a small group inside a big population, that has ASD.
To do that Recall is a suitable metric, expressed as:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

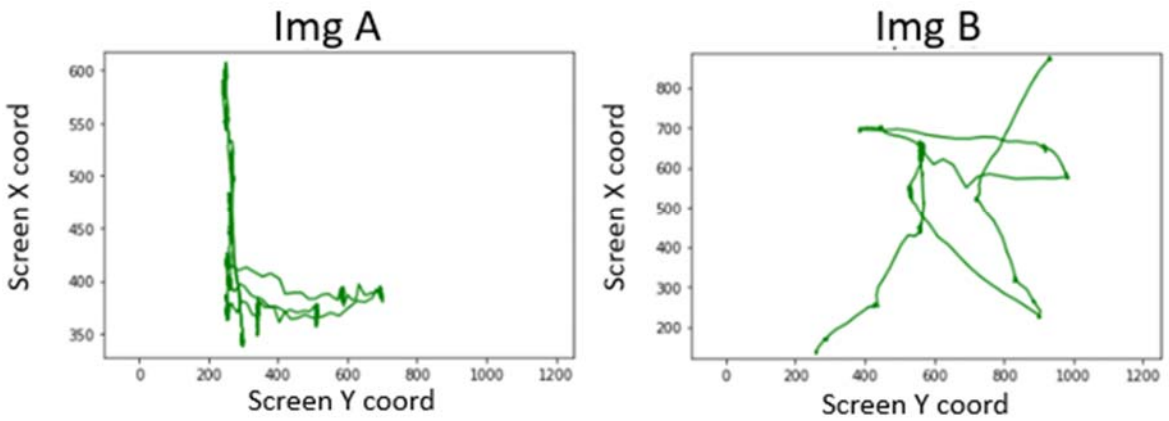
This metrics refers to the percentage of total relevant results correctly classified by the model, this is the important parameter in our case

## II. Analysis

### Data Exploration

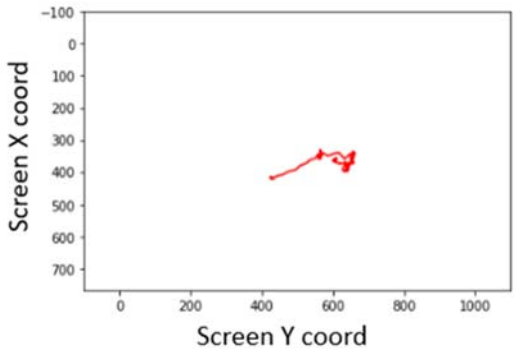
The image sequence consists in 22 images, with 4 second of exposition each, with crosses before, between and after the images, with a duration of 2 seconds per cross
The sequence is: Cross_1, image_1, cross_2, image_2,.....cross_22, image_22, cross_23.
The gaze is capture by a Tobii Eye tracker with a sample rate of 300 Hz, so we have a continuous temporal register for all images and crosses which consist in X and Y position of the gaze over the screen. We have 600 data points for each cross, and 1200 points for each images.
So, in principle, we have 22+23 = 45 different gaze exploratory patters (gazeScreenPosition[posX, posY, t] for each individual, and our universe is composed by 10 ASD and 31 TD individuals

First, we need to split the data into segments corresponding to crosses and images. In the next picture we can see the track of the position of the gaze over two different images, combining x and y position.



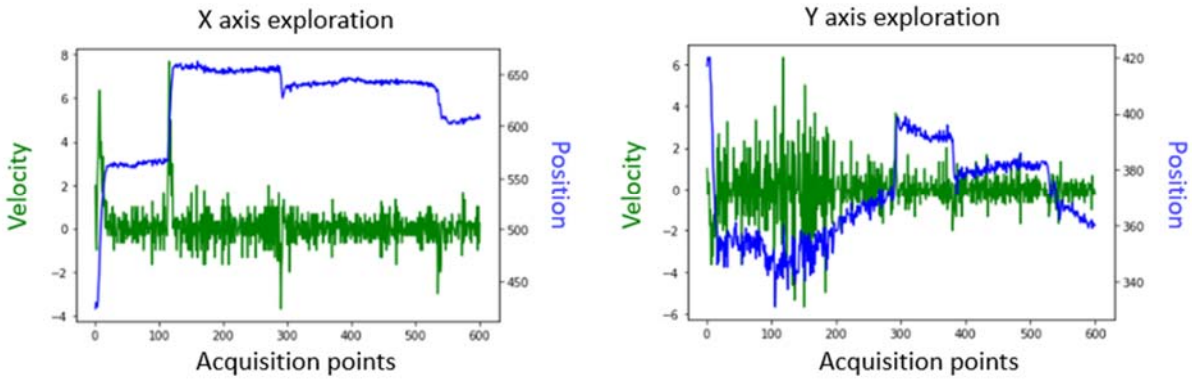

Exploratory behavior over two different images.

We can clearly appreciate that there are preferential spots of view, width rapid transitions between them. Those patters depends on the individual and the stimulus (image).
In the next picture, the gaze movement of a cross search.



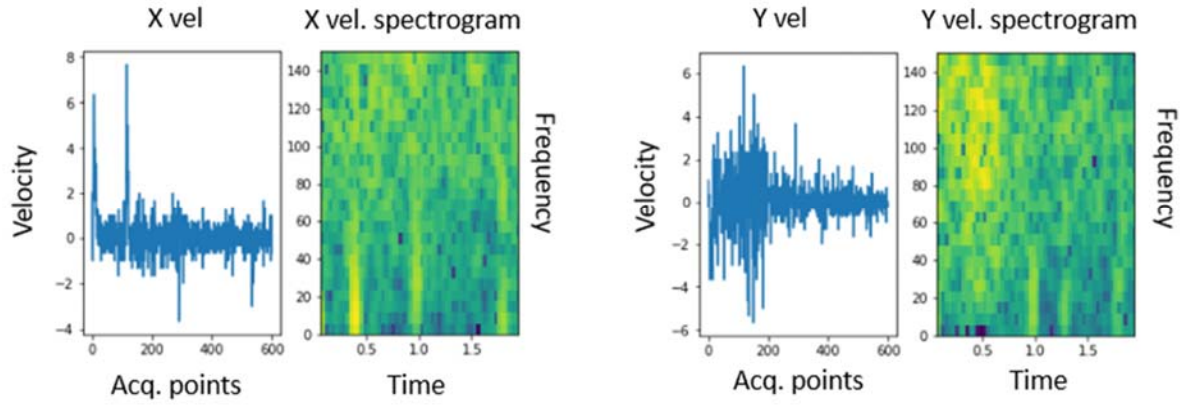
Gaze of a subject searching for the central cross.

Those images offer information of where the subject is seeing, but doesn´t contain temporal information, just spatial one.
In the next two figures we can see the positions (blue, right axis) and calculated velocity between adjacent points (green left axis, arbitrary unis) separated by axis X and Y, of the previous cross search data showed in the previous figures.
X axis:




Gaze position (blue) and corresponding velocities (green) over X and Y axis in the Cross search movement (previous figure)

In those graphs there exist temporal information of the ocular movement, but not the spatial one.
Our hypothesis said that in this movement pattern is where the information we need exist.
In order to use this velocity information with a convolutional network express the temporal variations of the velocity as a spectrogram, as we can see in the next picture



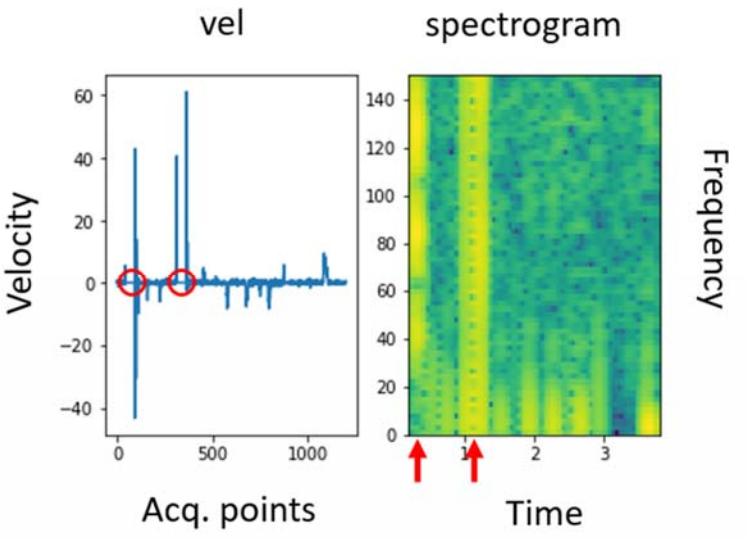

Velocities and corresponding spectrograms of the cross search gaze.

The spectrogram shows how velocity frequencies changes during the exploration, so it is a map of the exploration behavior.

We will use is to use both spectrograms from X and Y velocity, to feed a dual input CNN to make the classification.

**DATA PROBLEMS.**

We have detected an important problem in the data, we think mainly due to bad configuration of the experimental setup: there are gaps in the X and Y position segments that have no information. Filling those gaps is a challenging task, because it can´t be made just making an average of the adjacent positions. When we did that an artifact appears in the spectrogram images, as we can see in the next picture, which depicts a velocity derived from a filled position segment and it corresponding spectrogram.




Some artifacts appears in the spectrogram ( red arrows) due to the filled gaps in the raw data, which as consequences in the calculated velocity ( red circles)

The filled positions imply zero velocity (red circles) and abrupt transitions in the edges of the filling area. Due to this in the spectrogram appears artifacts (red arrows) in the form of massive frequency components.

Probably it can be avoided using a more complex method to fill the gaps, but it is out of our current scope. (this helped us to redesign experimental parameters)

Due to this fact, we have discarded all segments with data gaps, but it implies that our dataset is drastically reduced: only 15% of ASD data and 31% of the TD data is useful.

This problem have more implications beside the loss of useful data: We can only compares gaze behaviors under the same stimulus, I meant, we can compares-learns from data of subjects seeing the same image, but after the data selection we have just a few number of exploration for each image, not enough information to expect a CNN can learn.

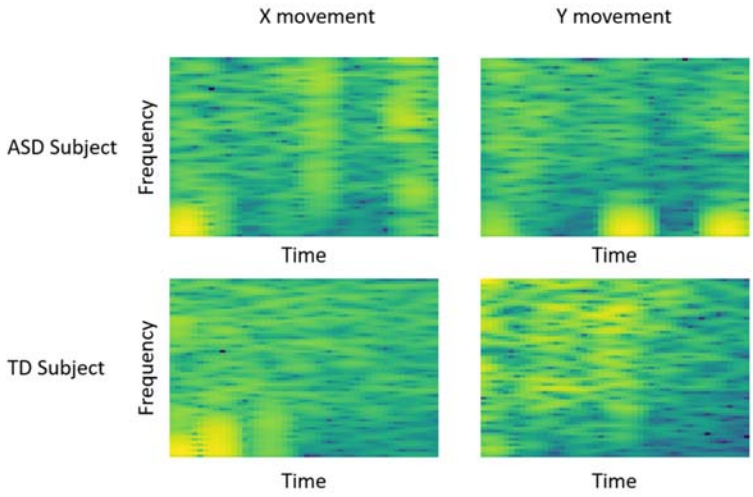
Fortunately, the cross is always the same comparable stimulus, and we have (we hope) enough crosses gaze exploration data to try the classification over them.

The data set is, because of that reason, composed by exploratory patterns of cross search images.

**Exploratory Visualization**

We use the gaze behavior in the crosses search to make our classification, the CNN will use two inputs, one for each axis of movement of the gaze, transformed into spectrograms of both X and Y velocities.

Example of X and Y cross search spectrograms of velocity from an ASD individual and TD individual are showed in the next figure




Spectrograms of the X and Y exploration velocities in the cross search for an ASD subject and a TD subject.

The spectrograms represent in an image the information of when and how the eye is moving, so it have the information of the gaze behavior in a way that a CNN can use. The model has to learn to classify individuals as ASD or TD based in this information.

**Algorithms and Techniques**

In the capstone proposal we suggest that the use of CNN´s as classifier, those are state of the art algorithms for image processing tasks, due to how the information is processed in the called convolutional and pooling layers, that is much more efficient and with less weight to adjust than in MultiLayer Perceptron (MLP) networks.
The convolutional layer explores the existence of several kernels ( aka patterns) of reduced size ( as small as a 3x3 matrix) over the entire image, using a moving convolution of the kernels over the image. Kernels are self-adjusted, in the training process, to the image characteristic, so the CNN

"learns" what kind of patterns needs to be in, or defines, the images corresponding to certain class.
Usually after a convolutional layer a Pooling layer is used, in order to reduce the spatial size of the representation, this also reduce the number of parameters needed in the network. Pooling layers offers some kind of compressed representation of the convolution result.
In this way the information that exists in the image is explored as a whole, not pixel by pixel, where the spatial related information contained in the image is preserved and used.

To analyse those temporal gaze patterns in order to classified the subjects as ASD or TD patients, and as first approach we thought in two possibilities:

**Approach A**: **Use raw gaze_screen_position**.

Make the classification ASD/TD from gazeScreenPosition[posX, posY, t]. This approach was discarded due to the lack of experience in temporal series analysis and the few available samples of data.

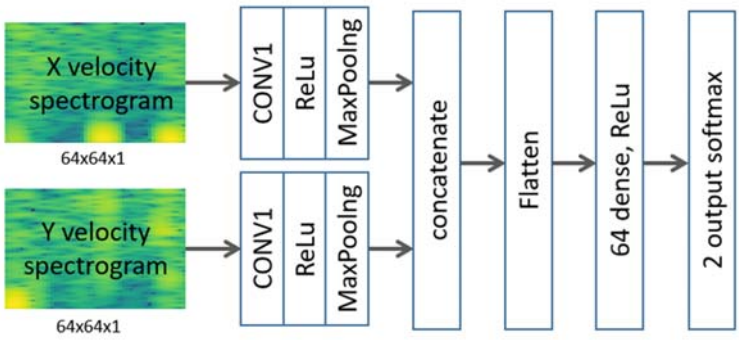
**Approach B: Use of derived data**

This one is the solution we have adopted: The temporal position, gazeScreenPosition[posX, posY, t] is transformed in gazeVelocities[velX, velY, t], which are expressed as spectrograms specX and specY, those feeds a dual input to the CNN described in the Methodology section. The needed of the same stimulus for all patients forced us to use just the gaze from Cross search segments. The spectrograms show, in just one image, how is the frequency of the eye movements exploring the stimulus, that is the biomarker we are looking for, and the CNN the patters that identifies ASD spectrograms from TD spectrograms.

As an alternative for future implementations (this is an active problem in our research group) Tobii`s software provides some gaze metrics ( fixation points, saccades,…) from which several multi-variate temporal series can be created and tested as inputs.

**Benchmark**

There not exist benchmarks for this problem, because of that, we created a benchmark with a naive CNN network, with the same architecture as our final model ( exposed later). In this case the CNN has only one Convolutional Layer with 16 kernels of 3x3.




Naive CNN used as benchmark

Over the test dataset this is the result ( ASD: label 0, TD: label 1)

```
test labels: [0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
pred labels: [1 0 0 0 0 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1]
```

The confusion matrix is as follows:

| | | Prediction | |
|---|---|---|---|
| | | ASD | TD |
| Truth | ASD | 5 | 3 |
| | TD | 3 | 39 |

Confusion matrix of the benchmark model

With a correct classification of 5 over 8 ASD subjects. This result is not so bad for that naive CNN.
The classification report from keras metrics gave us some clear points of comparison for our solution:

classification report

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.62 | 0.62 | 0.62 | 8 |
| 1 | 0.93 | 0.93 | 0.93 | 42 |
| micro avg | 0.88 | 0.88 | 0.88 | 50 |
| macro avg | 0.78 | 0.78 | 0.78 | 50 |
| weighted avg | 0.88 | 0.88 | 0.88 | 50 |

Classification report of the benchmark model

## III. Methodology

### Data Preprocessing

In order to feed our CNN the data preprocessing has the following steps:

1. For each gaze temporal data of each patient we segment the data corresponding to crosses and images.
2. Discard images segments and retain cross segments
3. Discard segments with NAN values (due to fails in the acquisition process)
4. From the remaining data of screen positions gazeScreenPosition[posX, posY, t], calculate velocities, gazeVelocities[velX, velY, t], of the movement between temporal adjacent positions
5. From Velocities generate one spectrogram for each of the axis of movement, generate the spectrograms, specX and specY, inputs of our CNN
6. Spectrograms are processed to be an image of 64x64 pixels and 1-dimension deep

The problems of gaps in the initial data implies that our dataset is reduce a lot, finally we have just 43 cross data from ASD subjects and 289 from TD subjects.
Our training set is composed by:

- ASD: 35 samples
- TD: 177 samples.

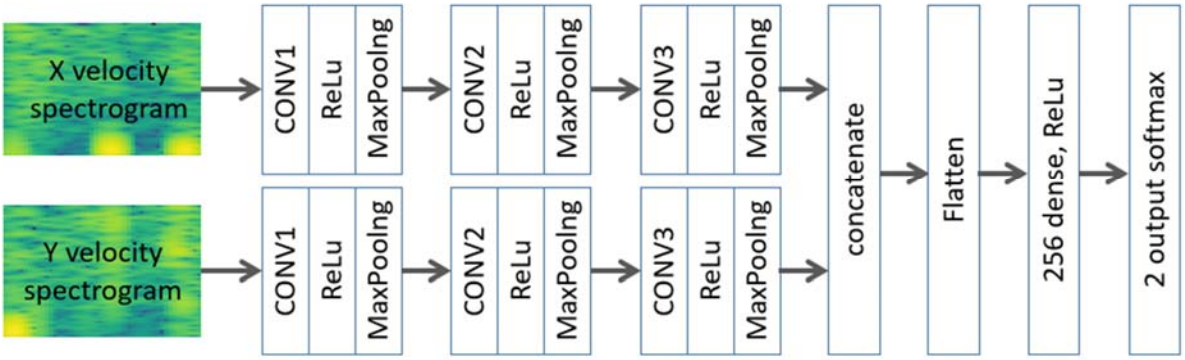
We cannot use data augmentation over ASD samples in order to make a more balanced training dataset. Any artificial change in the spectrograms generate arbitrary frequency components - the frequency components is the information we need- that will be not related with the patient behavior, so data augmentation just will generate bad data. In order to make the dataset balanced we used just 35 samples from TD patients.

The test set is composed by:

- ASD: 8 samples
- TD: 43 samples.

### Implementation

**CNN architecture.**
Our CNN consist in a dual CNN composed by 3 Convolutional layers each, fused trough a dense layer. After that, a perceptron layer with ReLu activation and two output softmax layer.




Depict of the dual input layer CNN used.

The two convolutional networks are twins, using 16, 32 and 64 kernels (conv1, conv2, conv3) of 3x3, ReLu activation and maxPooling.

To implement the network we have used keras functional API, the code that implements the net is showed below.
To create the convolutional layers we use a dropout of 0.5, in order to prevent early overfitting. In the training we have observed great tendency to overfit to TD cases.

```
def create_conv_layers( input_img):
    mod = (Conv2D(filters = 16, kernel_size= 3, padding ='same', activation ='relu', k
ernel_initializer = RandomNormal(mean=0.0, stddev=0.05, seed=None), input_shape=(image
Dim[0],imageDim[1],1)))(input_img)
    mod = (MaxPooling2D(pool_size = 2))(mod)
    mod = Dropout(0.5)(mod)

    mod = (Conv2D(filters = 32, kernel_size= 3, padding ='same', activation ='relu',ke
rnel_initializer = RandomNormal(mean=0.0, stddev=0.05, seed=None), input_shape=(imageD
im[0],imageDim[1],1)))(mod)
    mod = (MaxPooling2D(pool_size = 2))(mod)
    mod = Dropout(0.5)(mod)

    mod = (Conv2D(filters = 64, kernel_size= 3, padding ='same', activation ='relu',ke
rnel_initializer = RandomNormal(mean=0.0, stddev=0.05, seed=None), input_shape=(imageD
im[0],imageDim[1],1)))(mod)
    mod = (MaxPooling2D(pool_size = 2))(mod)
    mod = Dropout(0.5)(mod)
    return mod
```

The whole net is created using the create_conv_layers function:

```
inputShape = (imageDim[0],imageDim[1],1)

inputX = Input(shape = inputShape)
inputX_model = create_conv_layers(inputX)

inputY = Input(shape = (imageDim[0],imageDim[1],1))
inputY_model = create_conv_layers(inputY)

comb = concatenate([inputX_model,inputY_model])
flat = Flatten()(comb)
dense = Dense(256,activation = 'relu')(flat)
dense = Dropout(0.5)(dense)
output =Dense(2,activation = "softmax")(dense)

model = Model(inputs = [inputX, inputY], outputs = [output])
model.summary()
```

The resulting network summary:

| Layer (type) | Output Shape | Param # | Connected to |
|---|---|---|---|
| input_1 (InputLayer) | (None, 64, 64, 1) | 0 | |
| input_2 (InputLayer) | (None, 64, 64, 1) | 0 | |
| conv2d_1 (Conv2D) | (None, 64, 64, 16) | 160 | input_1[0][0] |
| conv2d_4 (Conv2D) | (None, 64, 64, 16) | 160 | input_2[0][0] |
| max_pooling2d_1 (MaxPooling2D) | (None, 32, 32, 16) | 0 | conv2d_1[0][0] |
| max_pooling2d_4 (MaxPooling2D) | (None, 32, 32, 16) | 0 | conv2d_4[0][0] |
| dropout_1 (Dropout) | (None, 32, 32, 16) | 0 | max_pooling2d_1[0][0] |
| dropout_4 (Dropout) | (None, 32, 32, 16) | 0 | max_pooling2d_4[0][0] |
| conv2d_2 (Conv2D) | (None, 32, 32, 32) | 4640 | dropout_1[0][0] |
| conv2d_5 (Conv2D) | (None, 32, 32, 32) | 4640 | dropout_4[0][0] |
| max_pooling2d_2 (MaxPooling2D) | (None, 16, 16, 32) | 0 | conv2d_2[0][0] |
| max_pooling2d_5 (MaxPooling2D) | (None, 16, 16, 32) | 0 | conv2d_5[0][0] |
| dropout_2 (Dropout) | (None, 16, 16, 32) | 0 | max_pooling2d_2[0][0] |
| dropout_5 (Dropout) | (None, 16, 16, 32) | 0 | max_pooling2d_5[0][0] |
| conv2d_3 (Conv2D) | (None, 16, 16, 64) | 18496 | dropout_2[0][0] |
| conv2d_6 (Conv2D) | (None, 16, 16, 64) | 18496 | dropout_5[0][0] |
| max_pooling2d_3 (MaxPooling2D) | (None, 8, 8, 64) | 0 | conv2d_3[0][0] |
| max_pooling2d_6 (MaxPooling2D) | (None, 8, 8, 64) | 0 | conv2d_6[0][0] |
| dropout_3 (Dropout) | (None, 8, 8, 64) | 0 | max_pooling2d_3[0][0] |
| dropout_6 (Dropout) | (None, 8, 8, 64) | 0 | max_pooling2d_6[0][0] |
| concatenate_1 (Concatenate) | (None, 8, 8, 128) | 0 | dropout_3[0][0]<br>dropout_6[0][0] |
| flatten_1 (Flatten) | (None, 8192) | 0 | concatenate_1[0][0] |
| dense_1 (Dense) | (None, 256) | 2097408 | flatten_1[0][0] |
| dropout_7 (Dropout) | (None, 256) | 0 | dense_1[0][0] |
| dense_2 (Dense) | (None, 2) | 514 | dropout_7[0][0] |

Total params: 2,144,514
Trainable params: 2,144,514
Non-trainable params: 0

We ended with more of two millions of parameters to train.
As compiling options we had use:
Optimizer: Adam, mainly because our reduced data set, Adam is proved to be well suited to sparse gradients

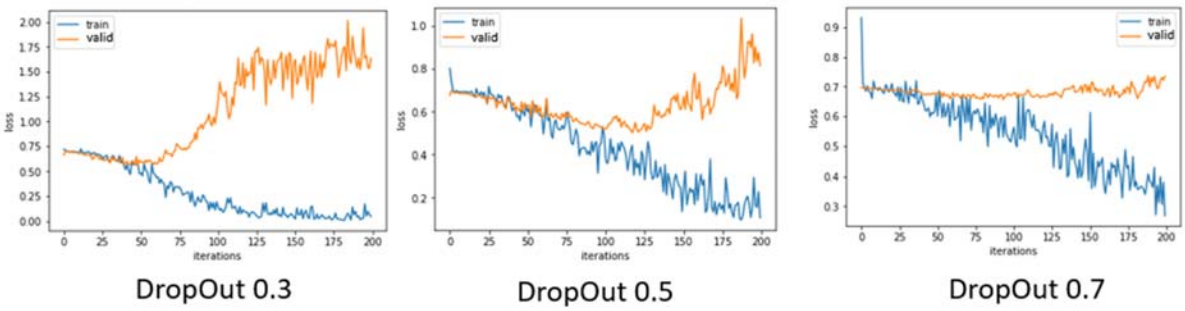
loss: categorical_crossentropy. Appropriate for binary classification.

In the coding process there was no important issues or complications. The main problem was the low quality of the data reflected as the existence of non-documented NAN values in the temporal stream. We had to implement a simple NAN detector in order to accept, or not, each one of the data. Besides that, my lack of experience in python programming was a challenge, but it is not related with the problem itself.

**Refinement**

In the training process we have some problems due to overfitting over TD subjects, we think mainly due to the initial unbalanced dataset. The CNN classified all subjects as TD ones.
To avoid this we have reduced the TD training samples to the same number of ASD samples, having a training set of 70 samples (35+35), using 60 as train and 10 as validation.

Still overfitting with the balanced dataset, we explore the Dropout of the convolutional layers.




Training and validation losses over the dropout exploration

With a Dropout 0.3 the CNN overfits to early, with 0.7 there are no evolution in the validation, we choose 0.5 as dropout probability.

In the training process we have use a three steps training, of 25 epochs and batch size of 5, with reloading the best weights in between each step.

# IV. Results

**Model Evaluation and Validation**

After the training the model, we have obtained the following result over the test set. (ASD label: 0, TD label: 1

```
test labels: [0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
```

```
pred labels: [0 0 0 0 0 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 1]
```

The test dataset consists in 8 ASD samples and 43 TD samples, all from individuals not included in the training

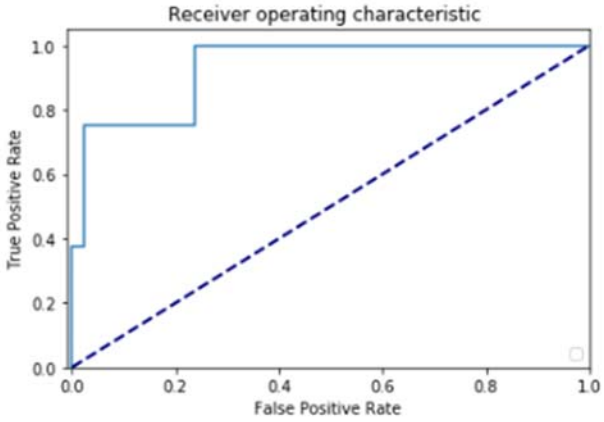
The classification report from keras metrics is as follows, has excellent results in the classification of the classes.

```
classification report
                precision    recall  f1-score   support

           0       0.70      0.88      0.78         8
           1       0.97      0.93      0.95        42

   micro avg       0.92      0.92      0.92        50
   macro avg       0.84      0.90      0.86        50
weighted avg       0.93      0.92      0.92        50
```

ROC curve with thresholds[ 0.  0.125 0.375 0.375 0.75  0.75  1. ]




The confusion matrix is as follow:

| | | Prediction | |
|---|---|---|---|
| | | ASD | TD |
| Truth | ASD | 7 | 1 |
| | TD | 3 | 39 |

Confusion Matrix of the final Model.

Recall is 7/(7+1) =0.875, which is a good outcome.

But the result of different training process are no consistent with this result, we had to train the CNN several times to obtain it. We think the main problem is the reduced size of the training dataset, offering those problems:

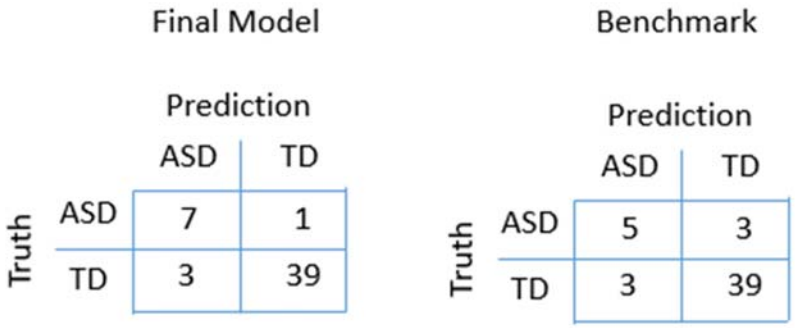
- most of the time the model overfit over one of the classes. We almost solved this making a balanced dataset and increasing the dropout.
- sometimes the model predicts with 100% of success, but confusing the classes, classifies all ASD as TD and all TD as ASD.

The dataset is too small to think that we can make a diagnosis based in this model, but is a starting point, hopping that in the near future we will arrange more cases to make a better training and a more trusted test result.

**Justification**
Comparing the results from the benchmark model we have improvements in all the metrics involved:

| Metric | Benchmark | | Model | |
|---|---|---|---|---|
| | ASD | TD | ASD | TD |
| Precission | 0.62 | 0.93 | 0.70 | 0.97 |
| recall | 0.62 | 0.93 | 0.88 | 0.93 |
| f1-score | 0.62 | 0.93 | 0.78 | 0.95 |

As we can see in the table, we have an improvement in all the metrics provided by keras metrics, so we can confirm that our solution is better than the benchmark.
And the confusion matrix comparison shows how our ASD detection is improved:

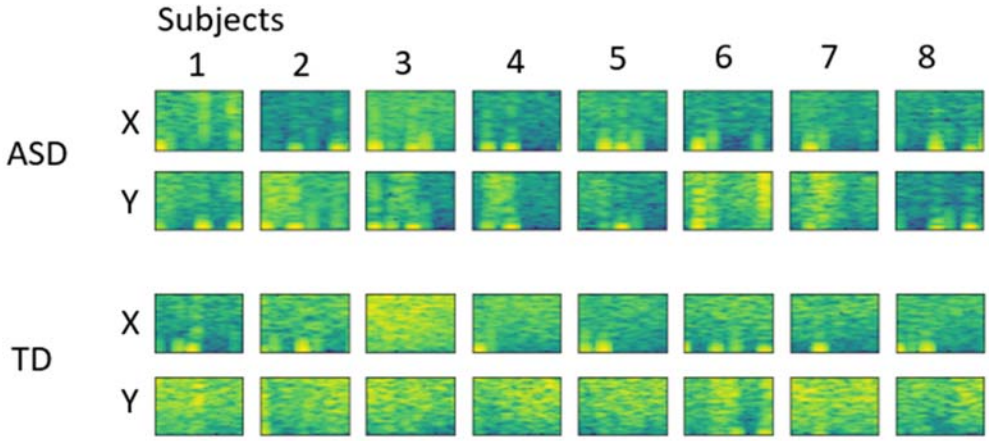



Confusion Matrix of the Benchmark and final Model.

Due to the small dataset we cannot confirm if the final model solve the proposed problem, but we think is a step forward, and encourage us to keep working in this line.

## V. Conclusion

### Free-Form Visualization




Pairs of spectrograms of movement of 8 ASD subjects and 8 TD subjects. The differences between them are what our classifier is observing.

The picture shows the spectrograms pairs corresponding to 8 of the ASD and TD subjects.
It is clearly appreciated the differences between both population members, mainly in the Y axis movement. ASD patients shows, in general, a more specific frequency movement location than TD ones. This differences are what our classifier is observing and use.

### Reflection

The process of making a automatic classification of individual Autism Spectra Disorder was (and still is) a challenging task, beside of the problems implied in the problem itself, this work allows as to have a better idea of our future experimental setups.
Data quality was the main problem we had, almost all data streams has corrupted sections. As result just 15% of ASD data and 33% of TD data was useful.
To feed the CNN we transform the raw data (gaze position in form of X, Y coordinates over a screen, and time) to an image, transforming positions into velocities and expressing velocities in form of spectrogram, which shows how different gaze movements occurs along the time of exploring.
As result of the bad data we cannot use the exploration behavior over images, too few examples over each image, but the cross search had the right characteristics: same stimulus for all individuals and enough different samples.
Beside the small dataset available, the results are promising, but we need to increase the number of samples in order to make a more robust training and test.

**Improvement**

In order to improve our results those are our future lines:

- We think that the use of spectrograms will be robust enough, so it will be part of our future model.
- There exist gaze characteristics, like saccades or fixation points, that can be used as inputs to the model, but not as an imaging. In order to incorporate them we will use the same network architecture, but expanding the number of input layers, probably using a combination of CNN with MLP networks as inputs.
- Our experimental setup needs to be revisited, for example, the sequence showed to subjects is too long, and that has a negative effect in the subject behaviour. Some threshold metrics need to be incorporated in order to have high quality dataset.