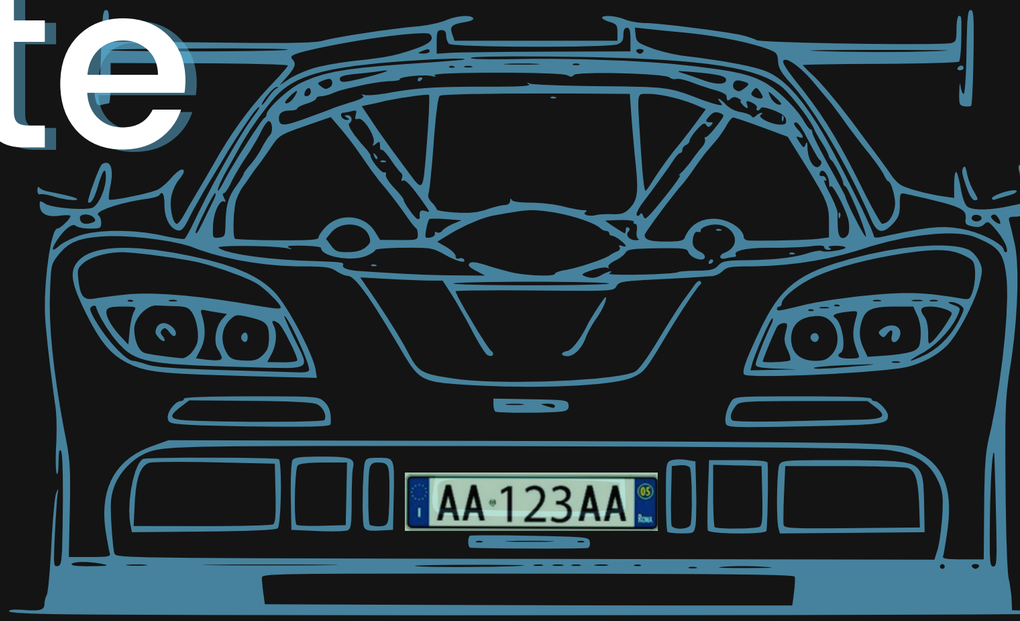
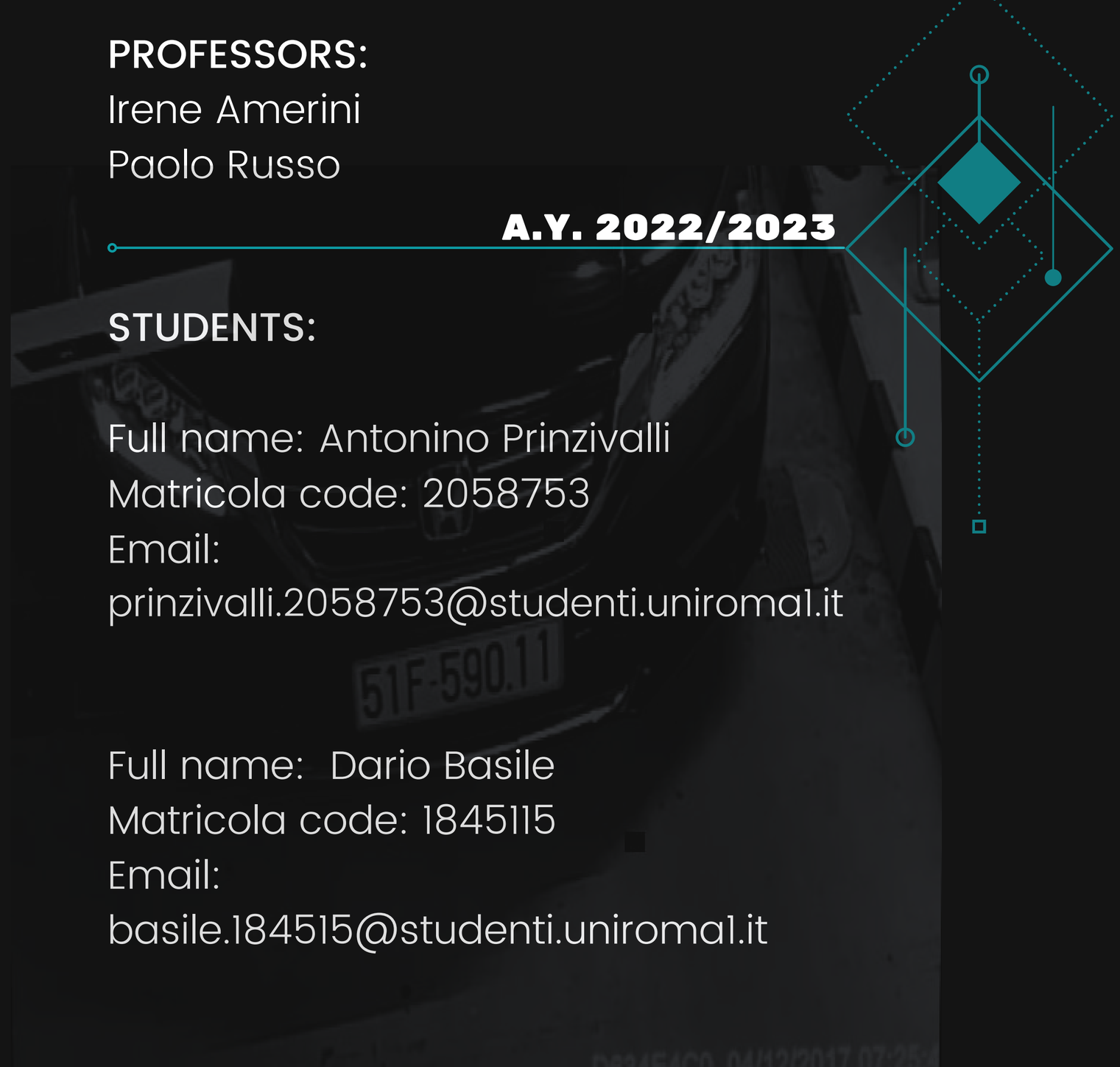
Vehicle license-plate recognition





PROFESSORS:

Irene Amerini

Paolo Russo

A.Y. 2022/2023

STUDENTS:

Full name: Antonino Prinzivalli

Matricola code: 2058753

Email:
prinzivalli.2058753@studenti.uniroma1.it

Full name: Dario Basile

Matricola code: 1845115

Email:
basile.184515@studenti.uniroma1.it

DESCRIPTION OF THE PROBLEM

The project aims to detect vehicle's license plates present in images in real world scenarios, and recognize the characters on them.

In order to do so, two convolutional neural networks are used: the first for detecting the plates and the second for recognizing the characters on them.

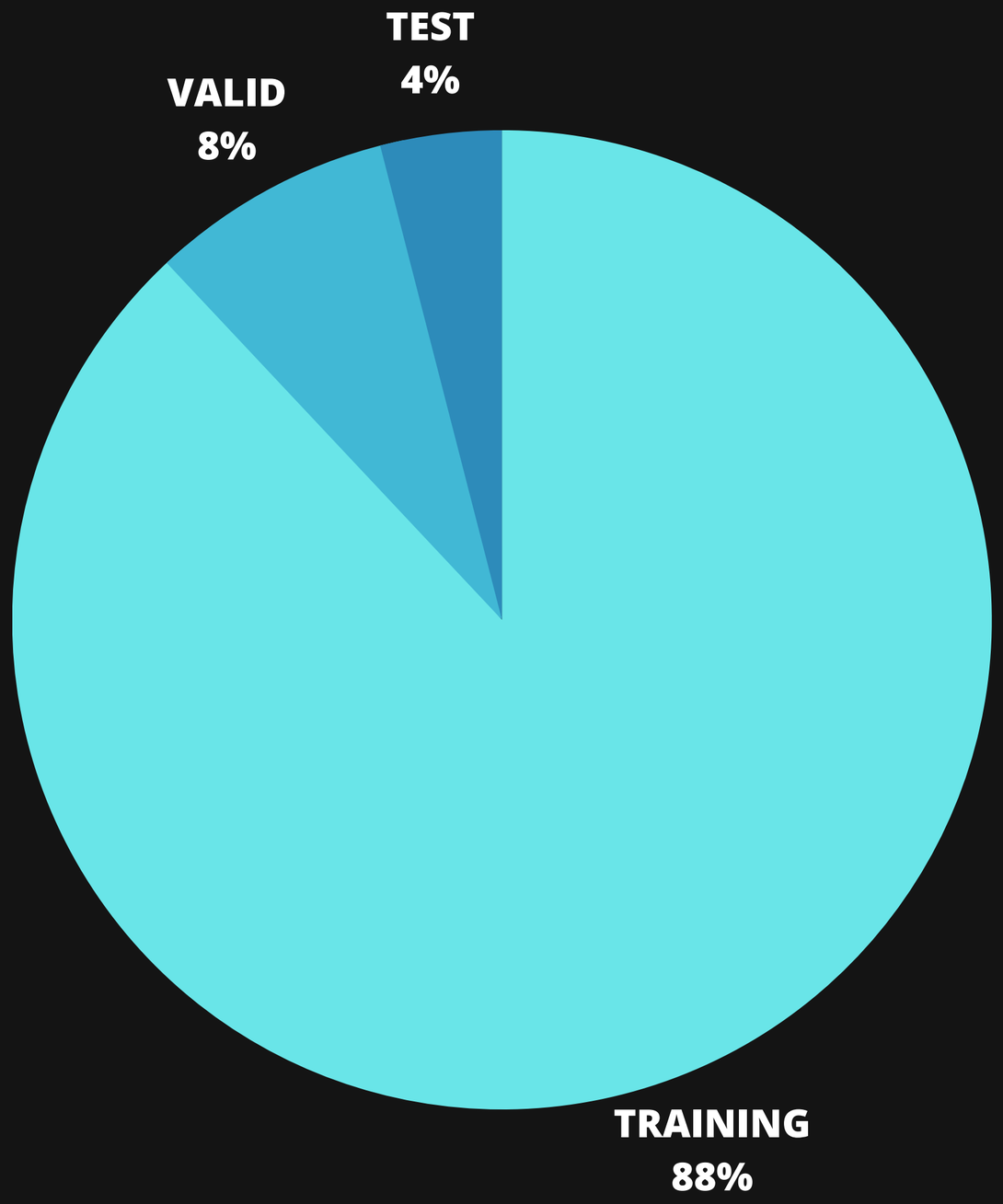
DATA SET EXAMPLES

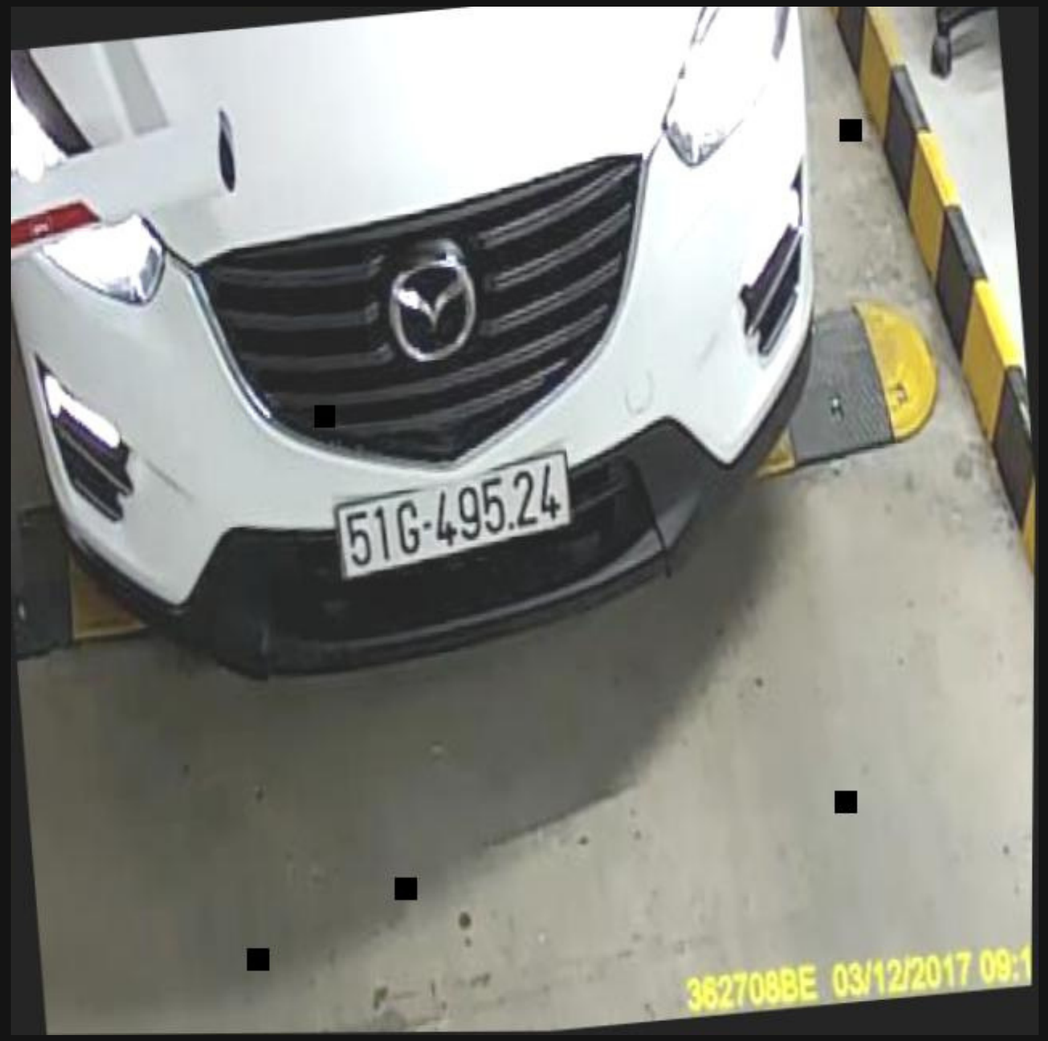
Data distribution:

21174 augmented images for training

2048 images for validation

1020 images for the test




51G-495.24
362708BE 03/12/2017 09:1


51F-590.11
D634E4C0 04/12/2017 07:25:

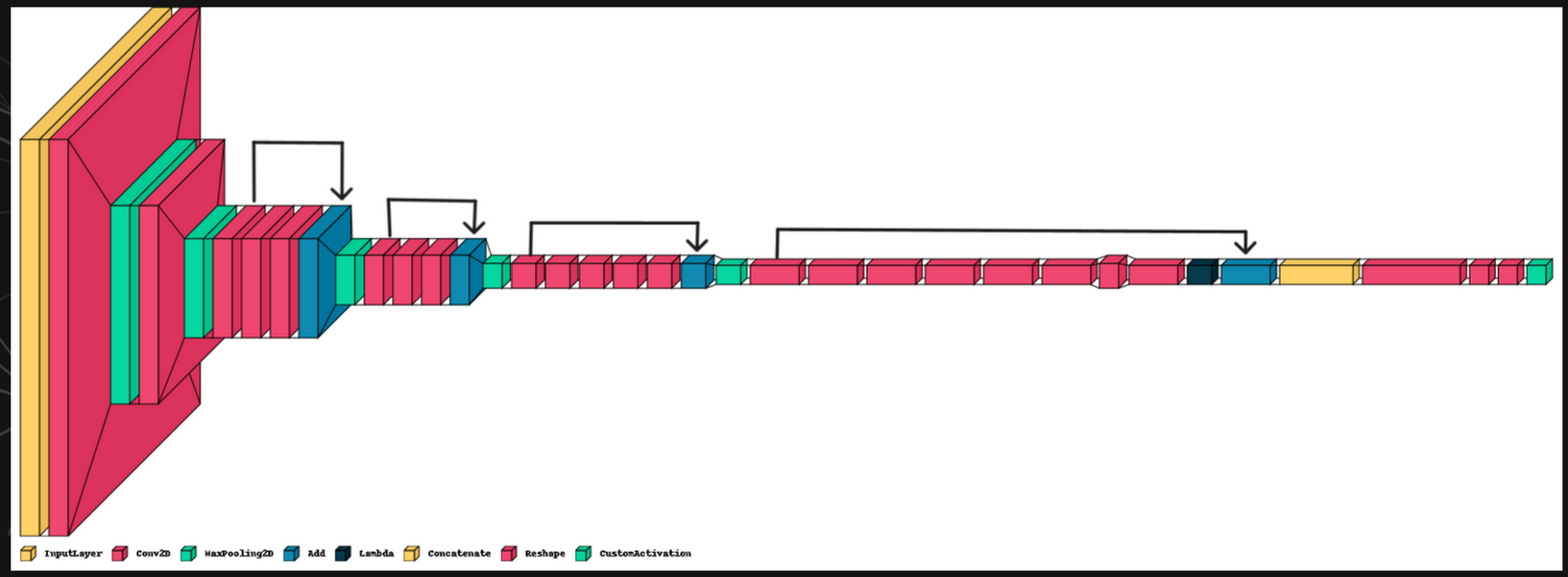
Type of transformations applied to the dataset:

Flip: Horizontal
Rotation: Between −45° and +45°
Shear: ±5° Horizontal,±5° Vertical
Hue: Between -15° and +15°
Saturation: Between -15% and +15%
Brightness: Between -30% and +30%
Gaussian Blur: (5,5) up to σ=2

CAR PLATE DECETION MODEL

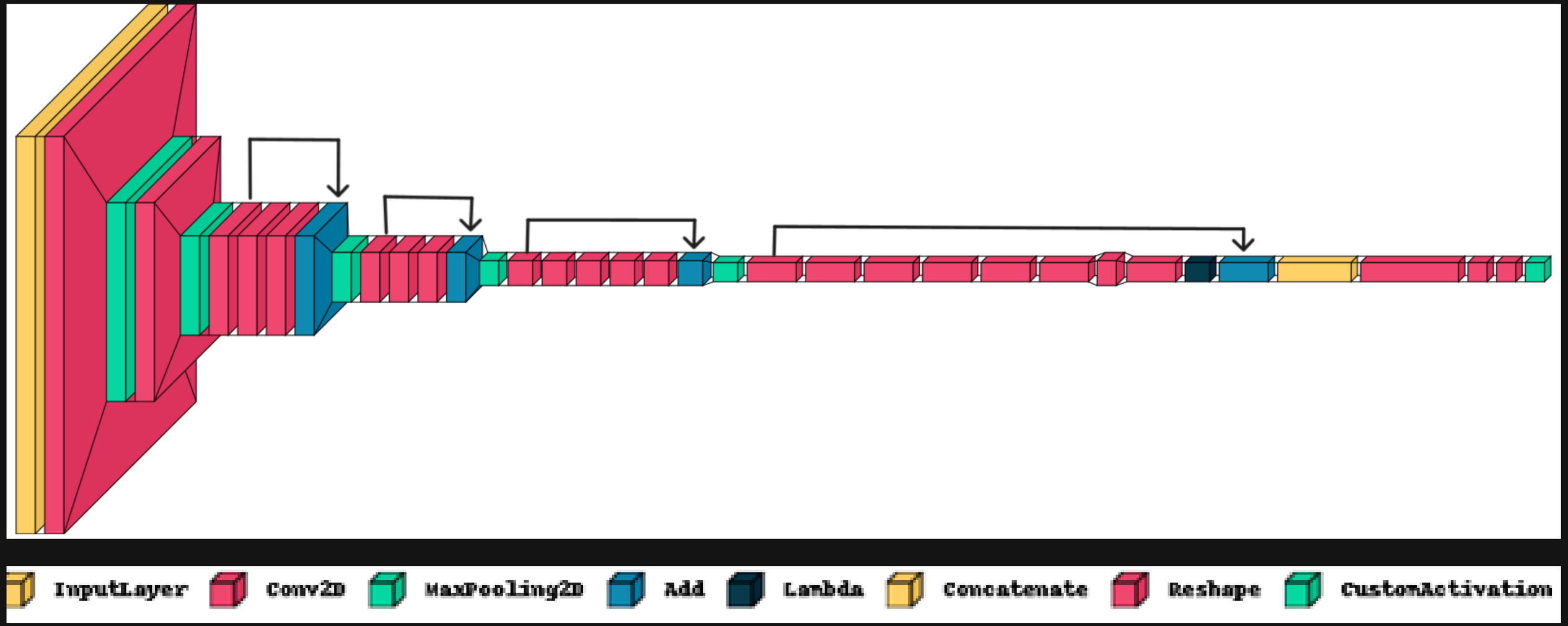
The car plate detection is a single stage model, which recognizes the presence of car plates in the image and localizes them.
The output will also contain a confidence score for the prediction.

How the car plate detetion model works



- 22 Layers deep
- Stack of Convloutional layers, Batch Normalization and Leaky Relu activation
- Use of residual connections
- Trained parameters: 20.214.009
- Input divided in 13x13 grid
- Use of 5 anchor boxes

Graphic representation



TRAINING

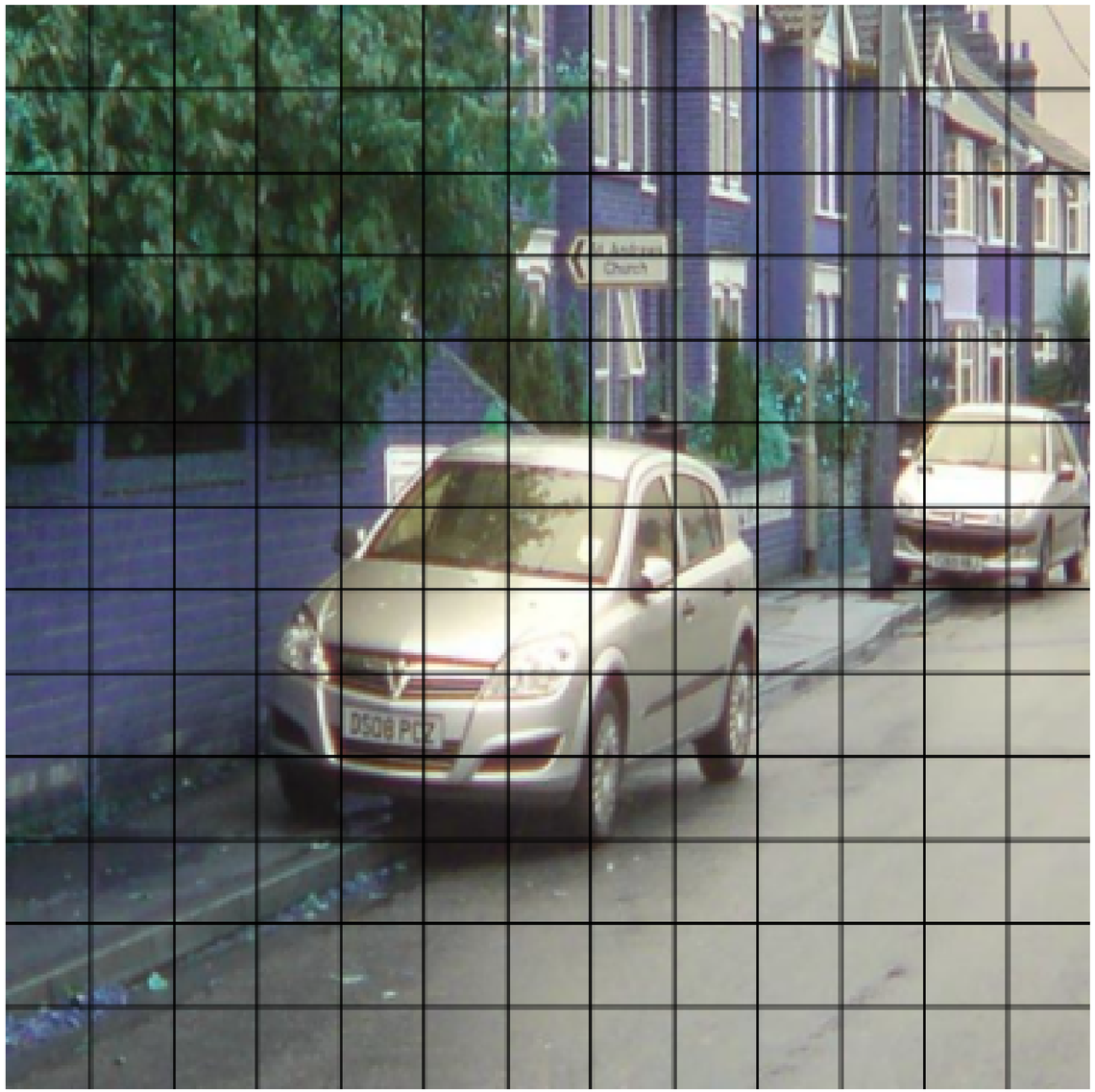
The image is divided in a 13x13 grid. Each cell will contain a label if a car plate is present in the cell.
Each prediction is a vector (for each cell and each anchor) containing the coordinates of the box and a confidence score. The predictions are shaped according to the anchors, which are computed

TRAINING

The image is divided in a 13x13 grid. Each cell will contain a label if a car plate is present inthe cell.
Each prediction is a vector (for each cell and each anchor) containing the coordinates of the box and a confidence score. The predictions are shaped according to the
anchors, which are computed

COMPOSITION OF EACH GRID

$$[X, Y, W, H, C]$$



13X13 GRID

LOSS FUNCTION

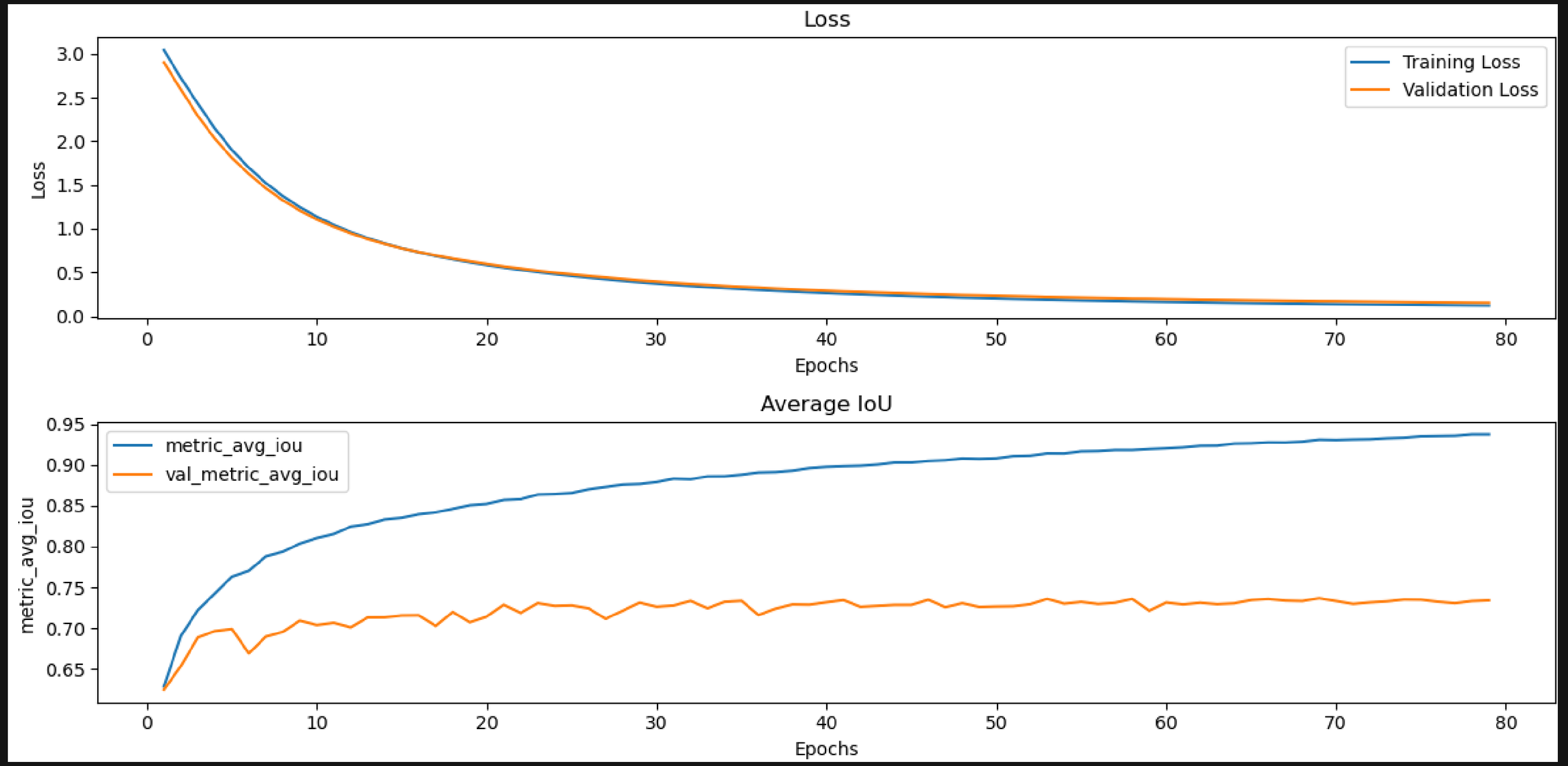
The loss is a combination of the coordinate loss and the confidence loss. The first is computed with MSE between predicted and ground truth coordinates, while the second is also a MSE between the the Intersection over Union

$$LOSS = \lambda_{coord} \frac{1}{N} \sum_{i=1}^{G^2} \sum_{j=1}^{B} \mathbb{I}_{i,j} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] +$$

$$\lambda_{coord} \frac{1}{N} \sum_{i=1}^{G^2} \sum_{j=1}^{B} \mathbb{I}_{i,j} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] +$$

$$\lambda_{is\,obj} \frac{1}{N} \sum_{i=1}^{G^2} \sum_{j=1}^{B} \mathbb{I}_{i,j} [(C_i - \hat{C}_i)^2] +$$

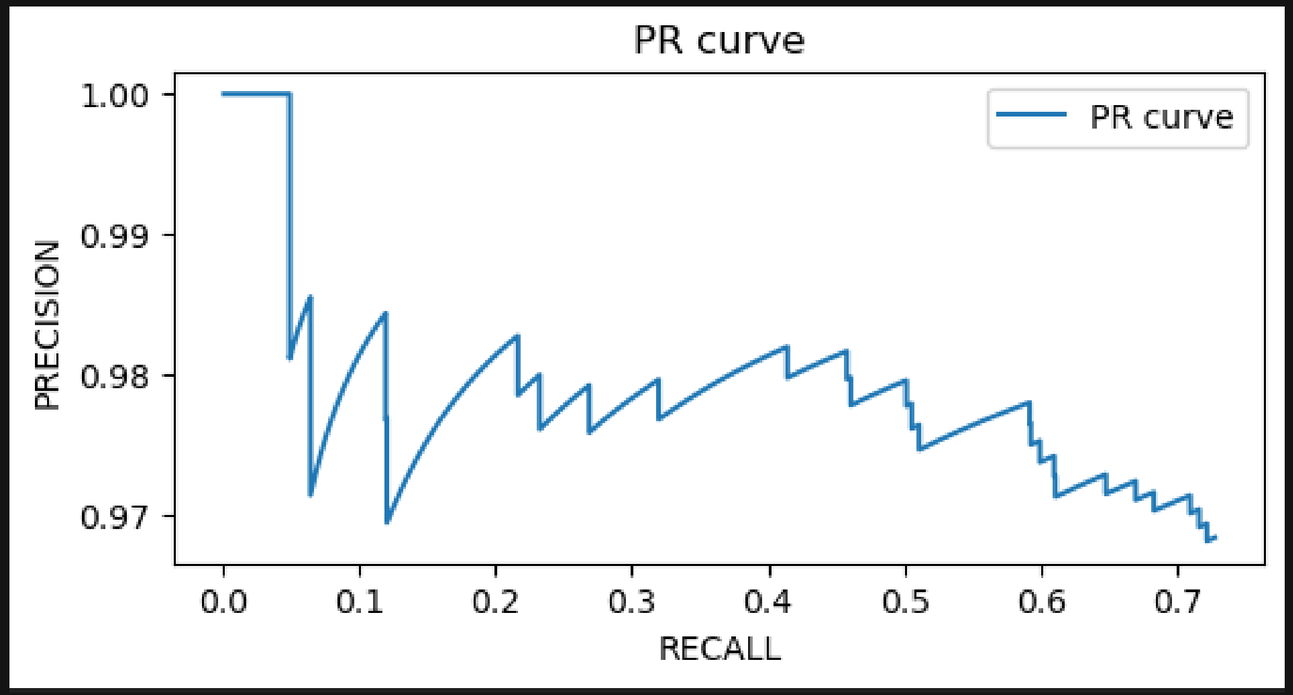
$$\lambda_{not\,obj} \frac{1}{N} \sum_{i=1}^{G^2} \sum_{j=1}^{B} (1 - \mathbb{I}_{i,j}) [(C_i - \hat{C}_i)^2]$$



We reached the lowest loss value almost after 80 epochs , with the use of ErlyStopping

Performance of the network for detecting the plate

```
Model performance:
   Precision: 0.9683944374209861
   Recall: 0.7274453941120608
   Mean IoU: 0.7241799071249554
   AP: 0.7109859978718198    (IoUthreshold  = 0.5)
   mAP: 0.46058237622879755
```



EVALUATION CAR PLATE DETECTOR NETWORK VS VGG16

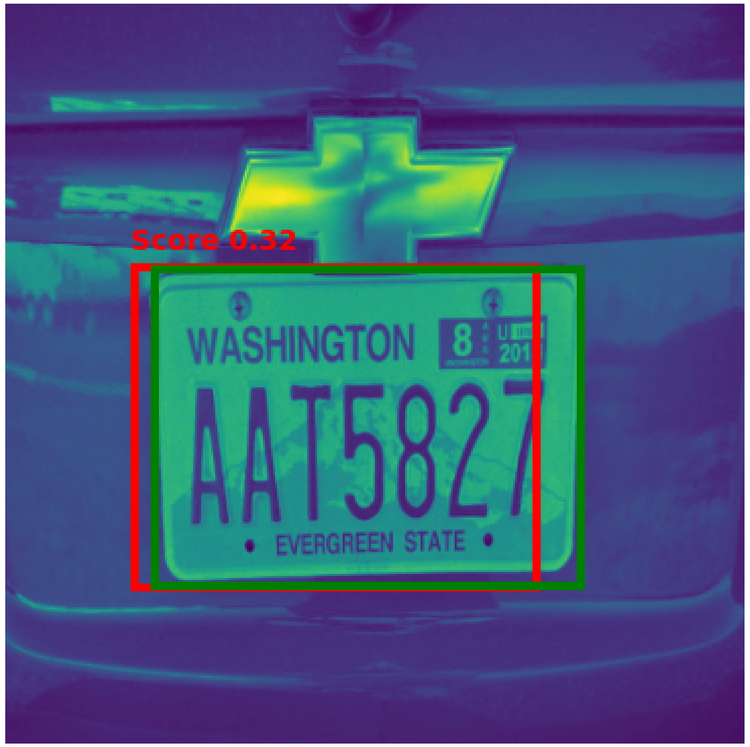
OUR MODEL

```
Model performance:
  Precision: 0.9683944374209861
  Recall: 0.7274453941120608
  Mean IoU: 0.7241799071249554
  AP: 0.7109859978718198    (IoUthreshold  = 0.5)
  mAP: 0.4605823762287975 5
```

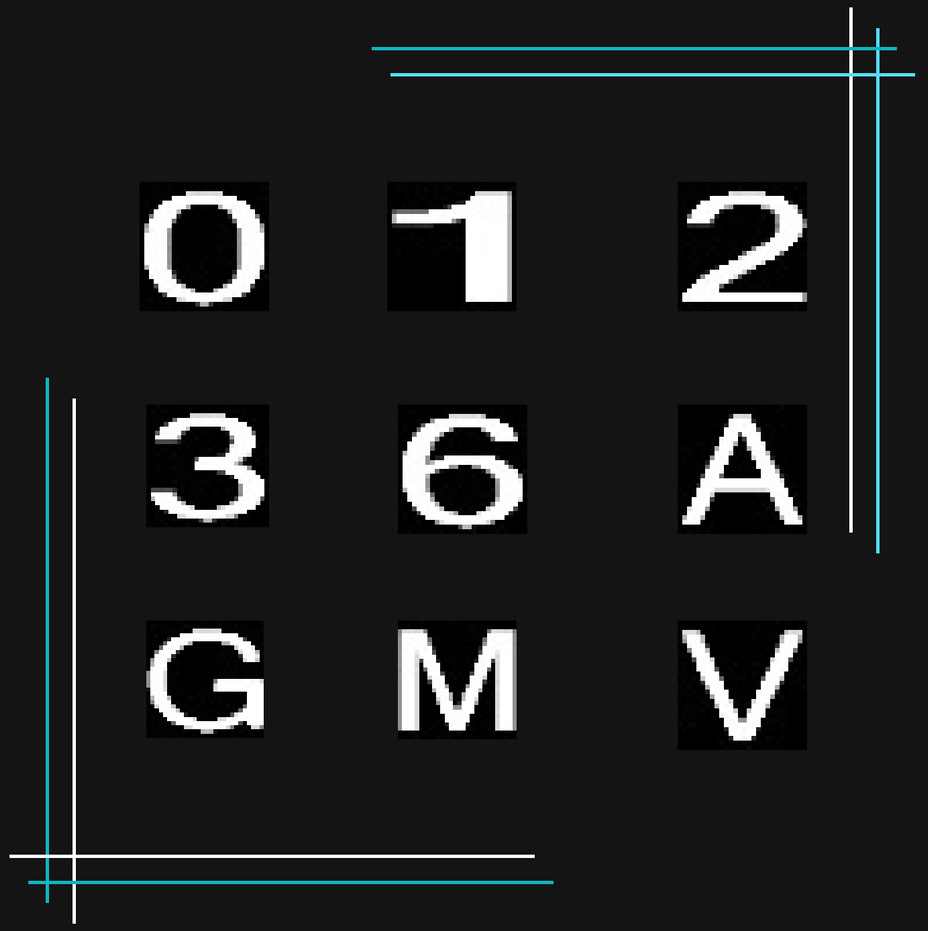
VGG16

```
"Precision": 0.844758064516129,
"Recall": 0.844758064516129,
"MeanIoU": "0.67912763",
"AveragePrecision": "0.7159928172873791",
"MeanAveragePrecision": "0.48384658"
```

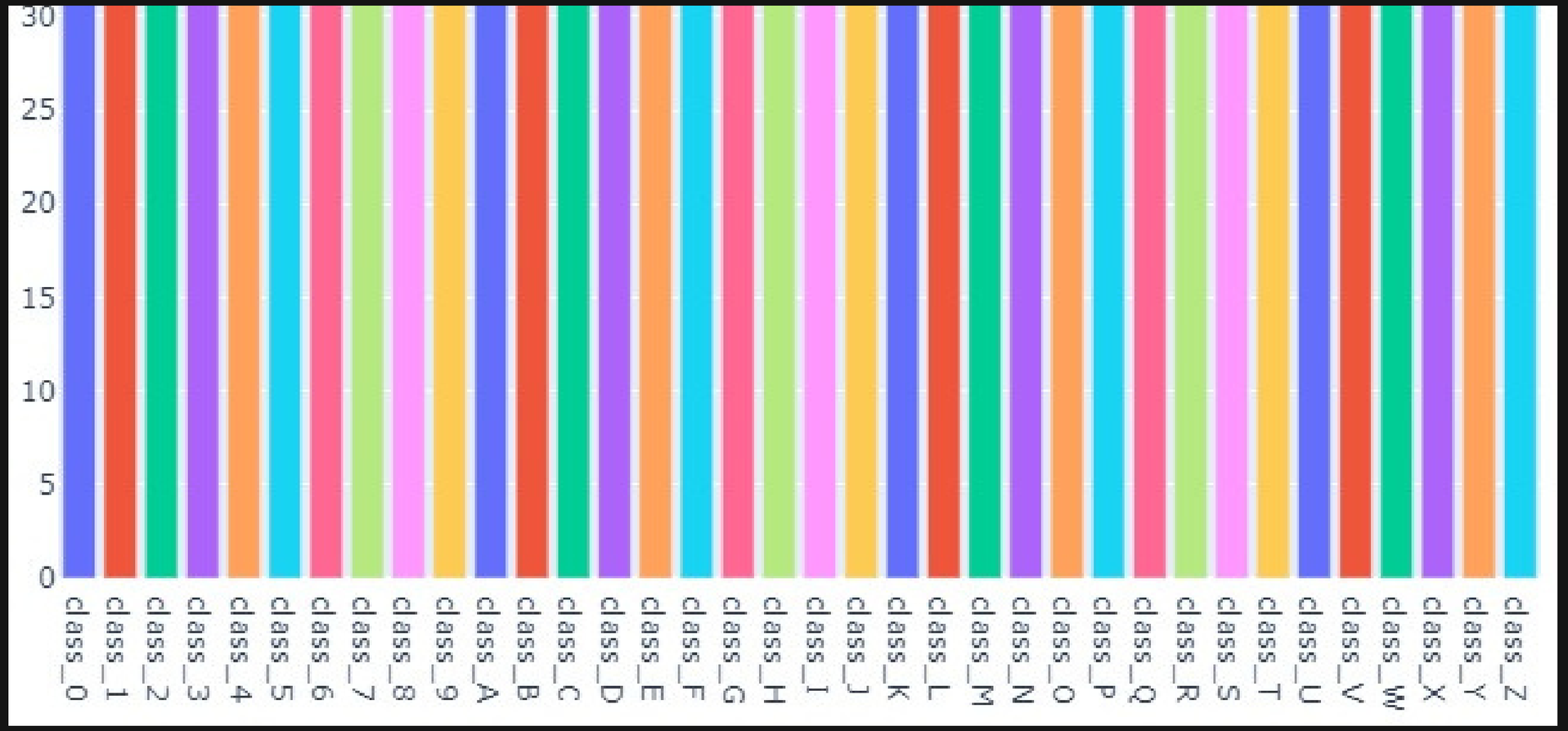
SOME EXAMPLES OF THE PREDICTION



data set for the characters recognition network



DISTRIBUTION OF THE DATASET



THERE ARE 36 CLASSES, WITH 30 IMAGES FOR EACH CLASS, OF WHICH 24 ARE FOR THE TRAINING AND 6 FOR THE TEST.

TYPE OF TRANSFORMATION APPLIED TO THE DATA SET:

- rescale = 1./ 255,
- zoom_range = 0.2,
- rotation_range = 45,
- brightness_range = (0.2, 0.8),
- width_shift_range = 0.1,
- height_shift_range = 0.1

Performing some image processing on the License plate
type of transformation on the plate before the character recognition:

- resize the dimension
- convert the colored image to a grey scale
- from grey scale to binary image(instead of 0-255, we have a 0/1 image) using a threshold function
- eroding: to eliminate noisy pixel that have value 1 but should have the value of 0, and dilate, that does the opposite, both work by looking at the neighbours

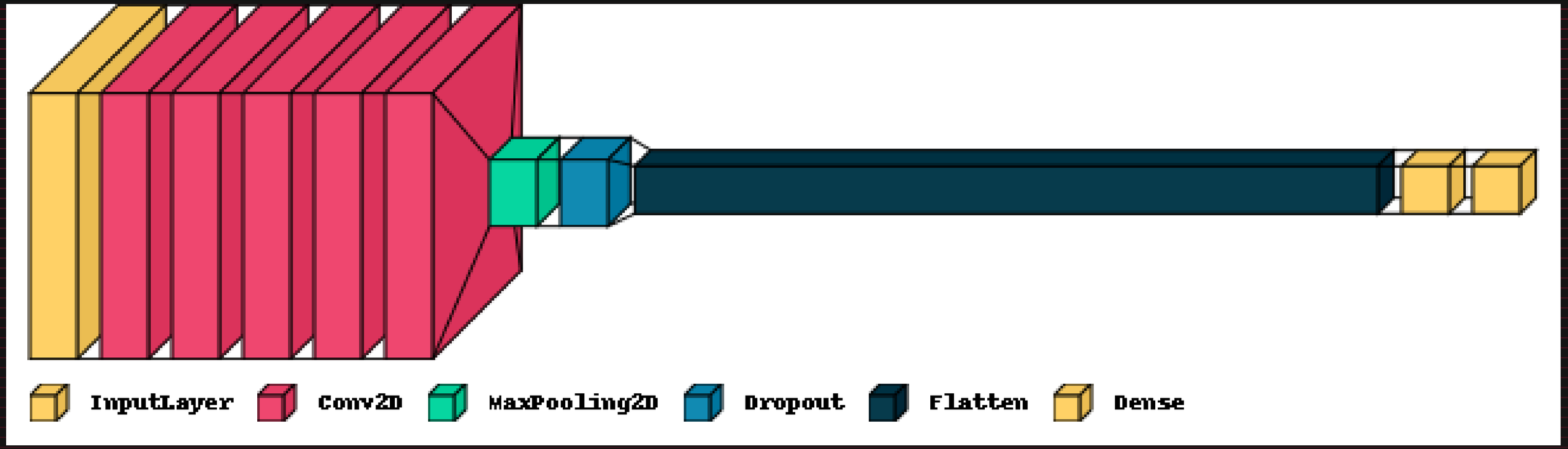
Segmenting the alphanumeric characters from the license plate.

We used: **cv2.boundingRect** to return all the rettangles for the contours in the image, but since it finds not only the rettangles related to the character, we used some constraints to take only rettangles with some dimension.

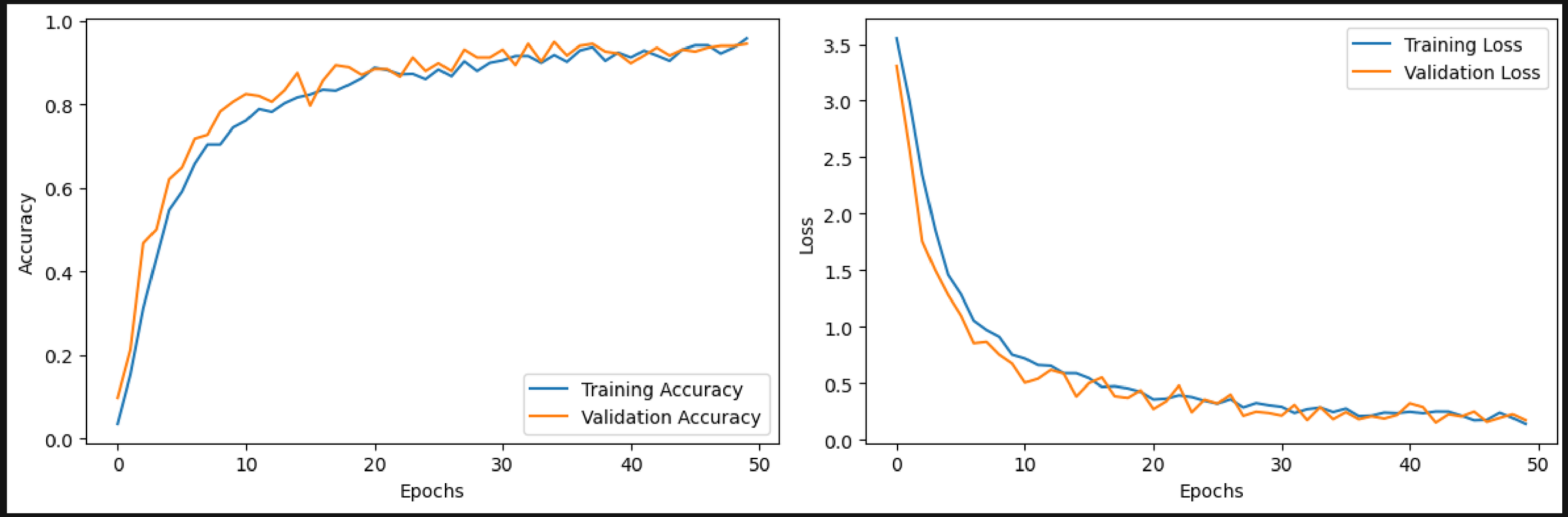
At the end we have a binary image with the extracted characters.

From this cropped plates obtained with the inital car image

TO THESE RESULTS


InputLayer
Conv2D
MaxPooling2D
Dropout
Flatten
Dense

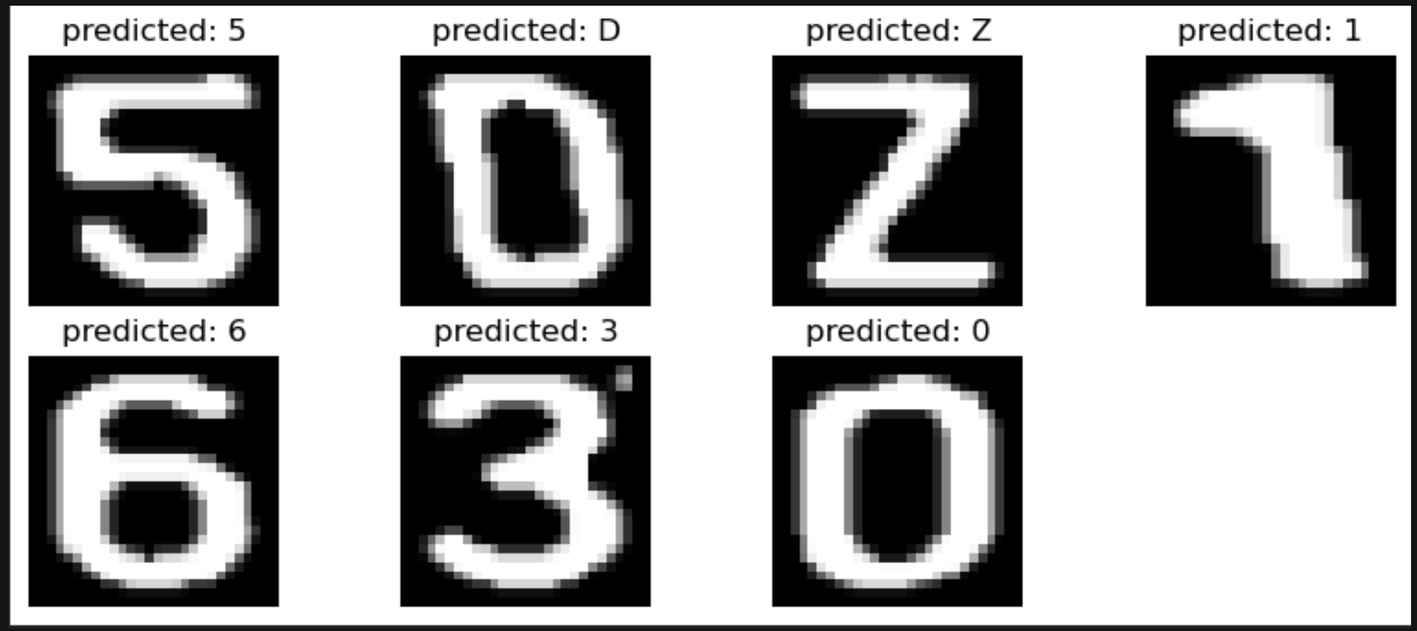
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_15 (Conv2D)           (None, 28, 28, 64)        1792

conv2d_16 (Conv2D)           (None, 28, 28, 16)        495632

conv2d_17 (Conv2D)           (None, 28, 28, 32)        131104

conv2d_18 (Conv2D)           (None, 28, 28, 64)        131136

conv2d_19 (Conv2D)           (None, 28, 28, 64)        65600

max_pooling2d_3 (MaxPooling   (None, 7, 7, 64)         0
2D)

dropout_3 (Dropout)          (None, 7, 7, 64)          0

flatten_3 (Flatten)          (None, 3136)              0

dense_6 (Dense)              (None, 128)               401536

dense_7 (Dense)              (None, 36)                4644

...
Total params: 1,231,444
Trainable params: 1,231,444
Non-trainable params: 0
```



The network needed almost 50 ephocs to complete the training

Given the segmented plate...

...the predicted output

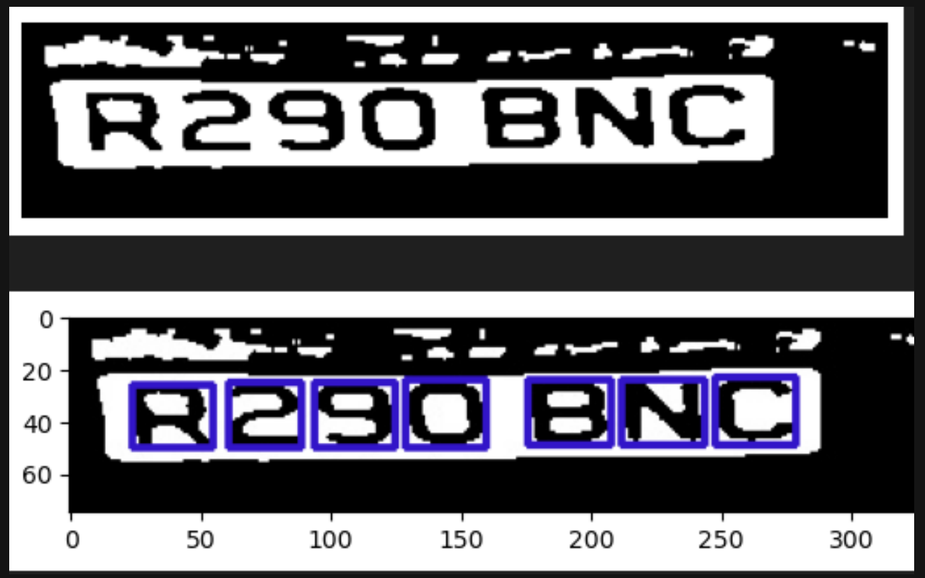


COMPLETE PROCESS

1) Load an image and preprocess it

2) Predict the location of the plates in the image, using the first model

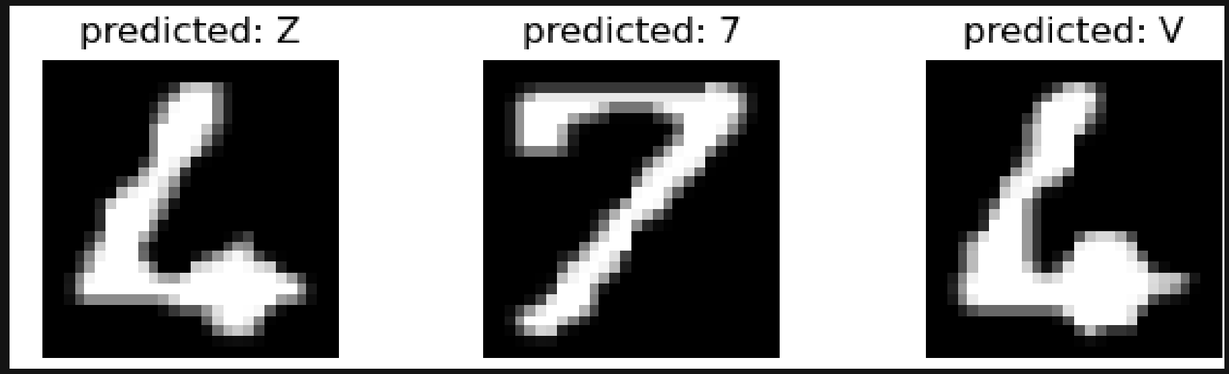
COMPLETE PROCESS



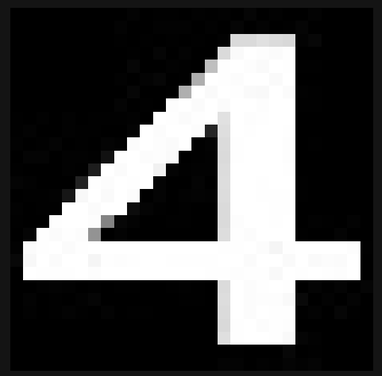
3)For each predicted plate compute segmentation for each character, using openCV functions

4) Each image of a character is passed to the second model which will predict its value, and the whole word will be formed by the their sequence

LIMITS AND FUTURE ENANCHEMENTS

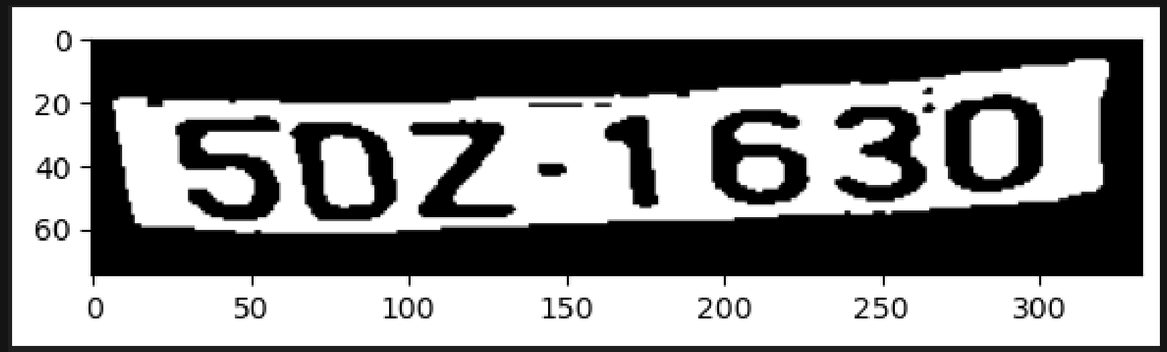


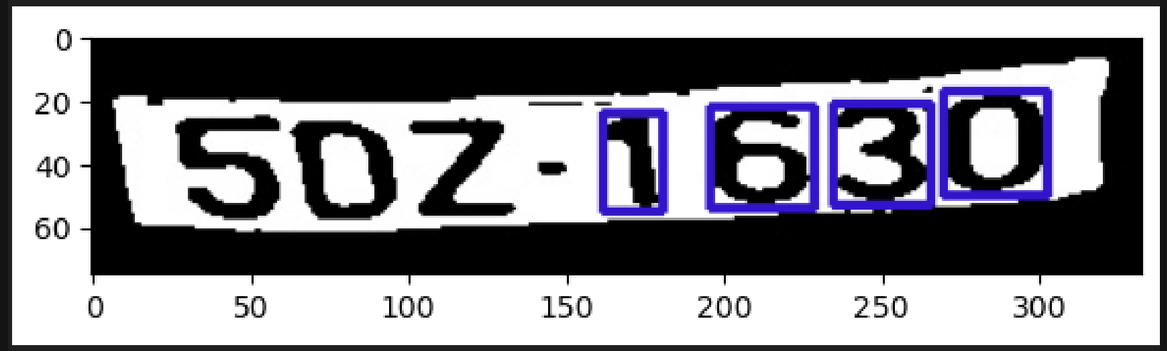
real numbers or characters of the plate didn't always match the type of the images in the data set for the recognition network



different type of numbers 4.
Above is the plate, on the left an image from the data set

Using a funcion to obtain rettangles it is not a robust choice, it depends on how we choice some paramiters





in the future for resolving this problem, we will implement a cnn for this task

PROBLEM OF THE CAR PLATE DETECTION MODEL

Confidence could get better results

Threshold to decide which prediction will be considered true positives


End