Trimble / Bilberry : AI Engineer technical exercise

Part 1 : project report

---

candidate : Rabah MOULAI

21 mars 2023

1 Introduction

The objective of this task is to train a neural network to classify images of roads and fields.

2 Methodology

The first task to be performed is to frame the problem in our case it's simple it's a supervised classification. Then Load and explore Data to get insights.

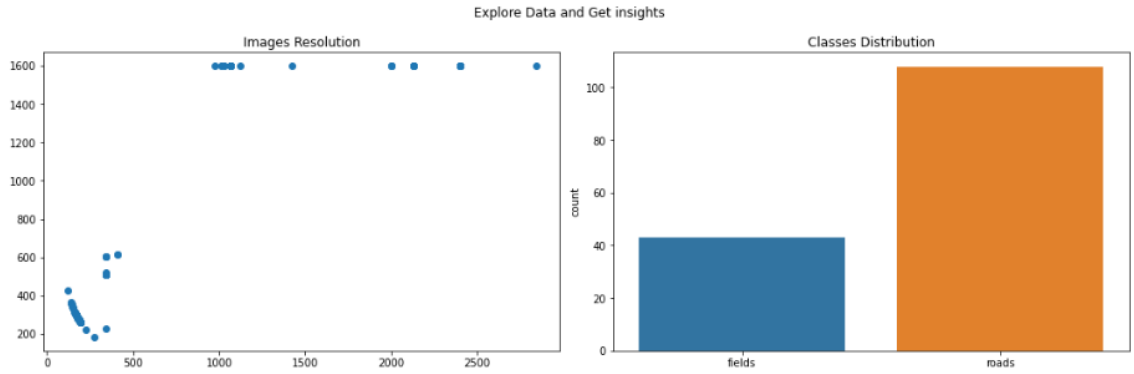

FIGURE 1 – Data Exploration

From this two plots we can see that the dataset is unbalanced and we can clearly see that images are clustered around height and width 550.

I decided to not remove the images with very high resolution to ensure that the model will generalize further i will resize the images to (124 x 124) to keep up the compute limitaions.

I will balance the dataset manually with class DatAugmentation (Oversampling the minority class). A small rotation is performed to equalize the number of examples in the majority class. Before data augmentation some transformations are necessary I resize the images to (124,124) and normalize the dataset by dividing by 255 to convert the pixels in range [0,255] to range [0,1]

I choose this approach to balance the data set against other approaches like (Undersampling Oversampling) because the dataset is relatively small. Note that two images were mislabeled, I removed it from false class to not bias the model

2.1 Pre-Processing

As the test set is provided without labels, I chose to divide the training set into a Test Set to evaluate the model and a set for Training and Validation. (80% training 20%test) Note that the 10 images provided with the test set will be used as inferences for the prediction

Even after balancing our DataSet, the DataSet remains very small To improve this, I will use ImageDataGenerator from tf.keras to augment the images with different augmentation techniques like rotation, small Zoom and a horizontal flip.

Other transformations were considered, such as rotations, vertical flipping, and applying a grayscale filter. These were not chosen, however, as they produce changes to the images that compromise their defining features. I also chose not to apply grayscale filtering, because the color gives important clues to the classification (e.g. fields are likely to have more green, roads likely to have more gray).

However, the main benefit of using the Keras ImageDataGenerator class is that it is designed to provide real-time data augmentation. Meaning it is generating augmented images on the fly while the model is still in the training stage.

2.2 Architecture and Model Training

For the architecture i choose a CNN Architecture with three convolutional layers, each followed by a maxpooling, and then finally two fully connected layers followeed Drop out regularization. This architecture is inspired from the AlexNet Architecture but is a much simplified version with fewer layers.

Use of convolutional layers on images allows the neural network to extract higher-level information about the image, such as edges. Max pooling both reduces noise and reduce dimensionality, allowing for more robust and quick training. ReLU activation is used because it results in much faster training time as compared to logistic or tanh.

The model will never see the same data twice, but some of the images it sees are strongly similar. The data is correlated because it comes from a small number of base images. With Data Augmentation we can't produce new information, we can only remix existing information. This may not be enough to get rid of the overfitting completely. So we will use the Dropout.

Finally, sigmoid is used on the output to produce probabilities that sum to 1 over all possible classes.

For Training the model i used Binary cross entropy which compares each of the predicted probabilities to actual class output which can be either 0 or 1. It then calculates the score that penalizes the probabilities based on the distance from the expected value. That means how close or far from the actual value. The dam optimizer is also used during the training phase.

Experimentally, I found roughly 25 epochs was enough to sufficiently train the model

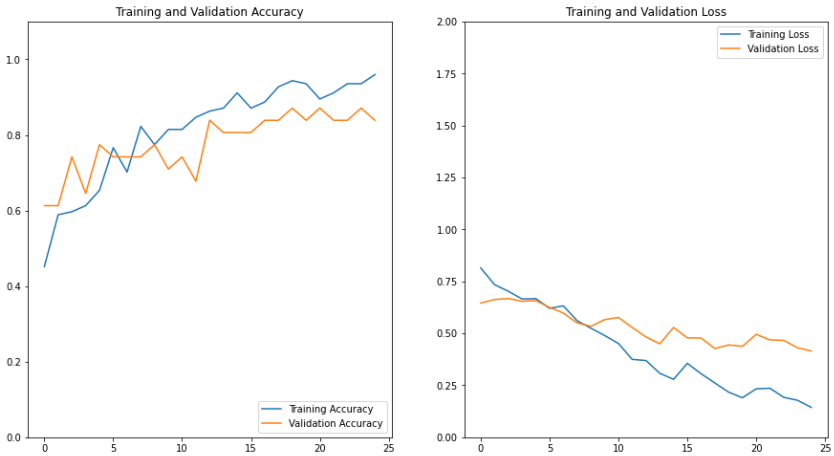

FIGURE 2 – Accuracy and Loss Curve

2.3 Evaluation Metrics and Results :

I evaluated our model with the test set but since it is a binary classifier we will not only evaluate the accuracy but also the ROC curve and the confusion matrix.

The accuracy on test set is between (85%to 97%). This variation is due to the random factor when increasing the data by the image generator.

A much better way to evaluate the performance of a classifier is to look at the confusion matrix. The general idea is to count the number of times instances of class A are classified as class B.

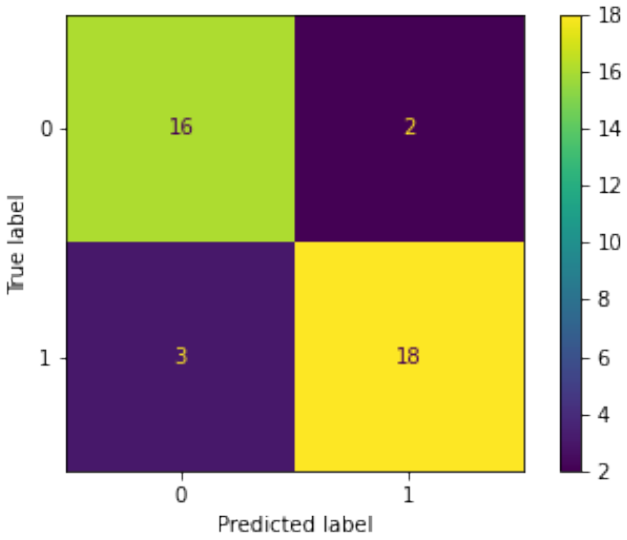

FIGURE 3 – Confusion Matrix

The confusion matrix show that only 5 of the 39 images are badly classified.

The receiver operating characteristic (ROC) curve is another common tool used with binary classifiers i found an AUC = 0.97.

2.4 Predictions :

I predicted the images provided with the Test set knowing that these images were not seen by the model neither in the training phase nor in the test (inferences)

FIGURE 4 – inferences Predictions

Here is the optimal case where the model is able to classify the 10 images. But, generally the neural network predicts correctly 8 to 10 of the 10 test images