TRAFFIC SIGN CLASSIFICATION

GERARD AGADA - MARCH 31, 2022

INTRODUCTION

- BCOMP, Computer Science - University of Guelph.
- BCOMM, Accounting - University of Guelph.
- Fintech Professional - Hedge Fund Accounting, Stock Market Indices & Portfolio Optimization.
- Interested in Machine Learning, Data Science, Autonomous Vehicles & Finance.

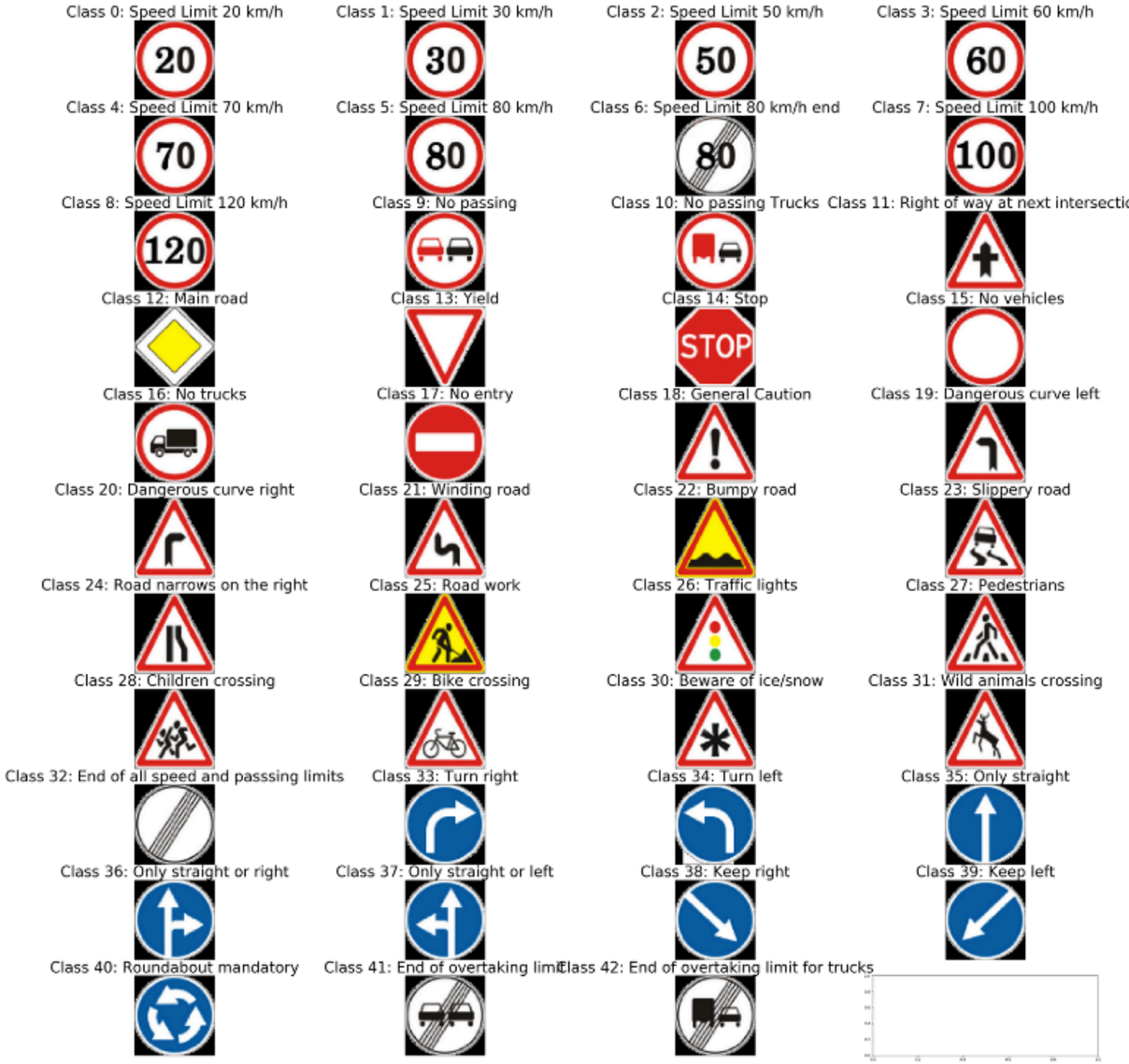
PROJECT WORKFLOW

- Business Case
- Importing Traffic Sign Data
- Data Exploration
- Pre-Processing The Images
- Model Development
- Model Evaluation
- Testing The Model
- Potential Challenges

BUSINESS CASE

- Autonomous transportation usage for buses, LRT's and taxis.
- Integration into autonomous vehicles for general consumer cars.
- Potential use cases for autonomous maintenance vehicles specifically garbage collection and snow removal.
- Autonomous delivery which can be facilitated by either drones or vehicles.

TRAFFIC SIGN DATA



DATA EXPLORATION

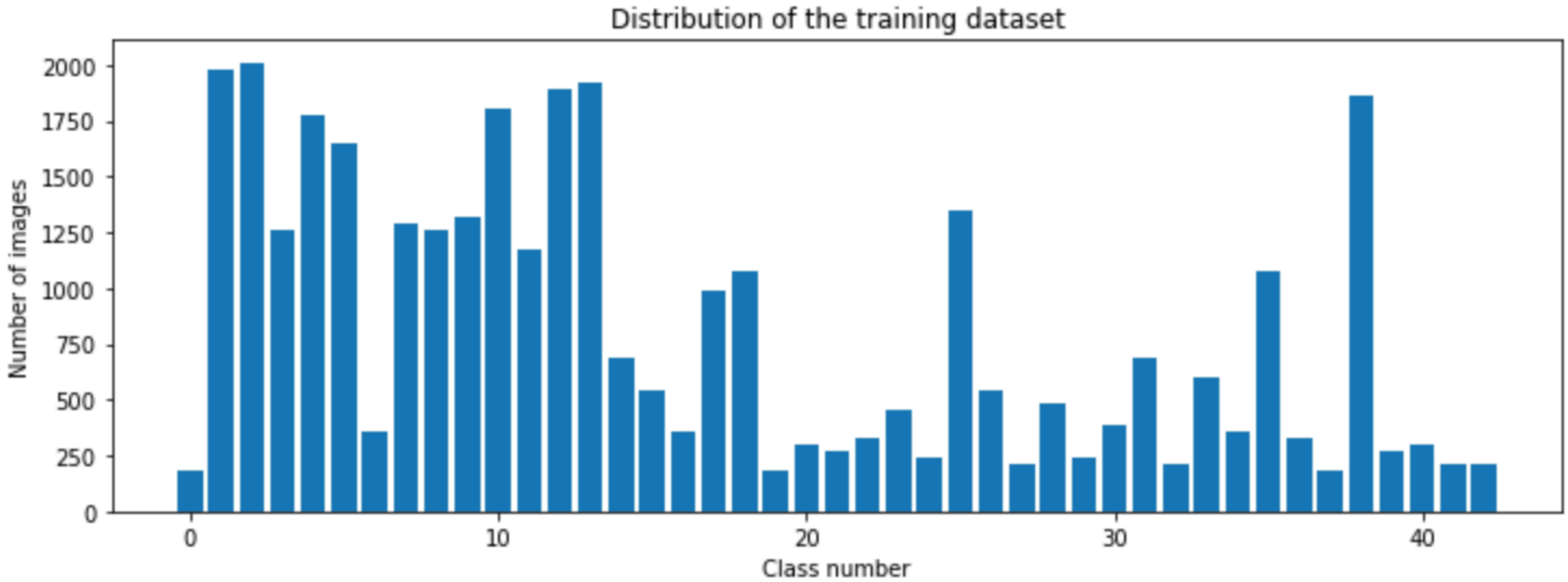
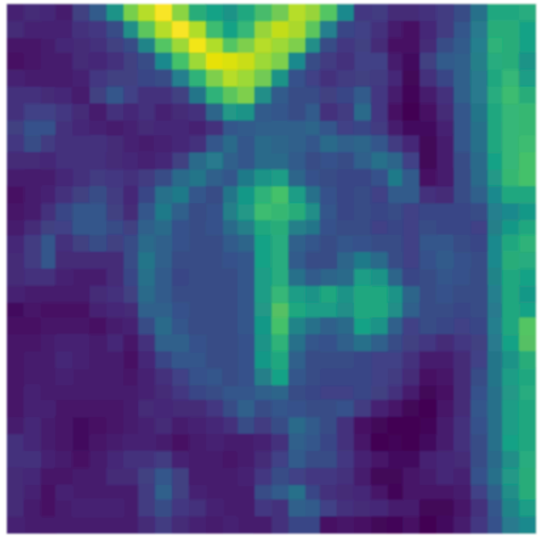
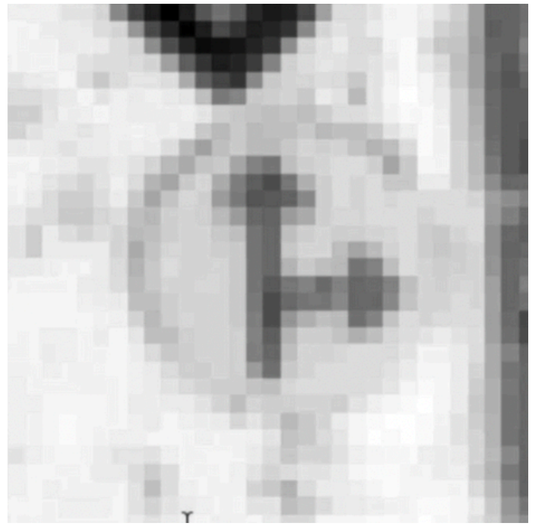


IMAGE PRE-PROCESSING

- We need to reduce an RGB image which is in the 3D space to a greyscale image which is the 2D space.

- With an image that is 2D, less computing power is required and our model can classify data more efficiently.





MODEL DEVELOPMENT

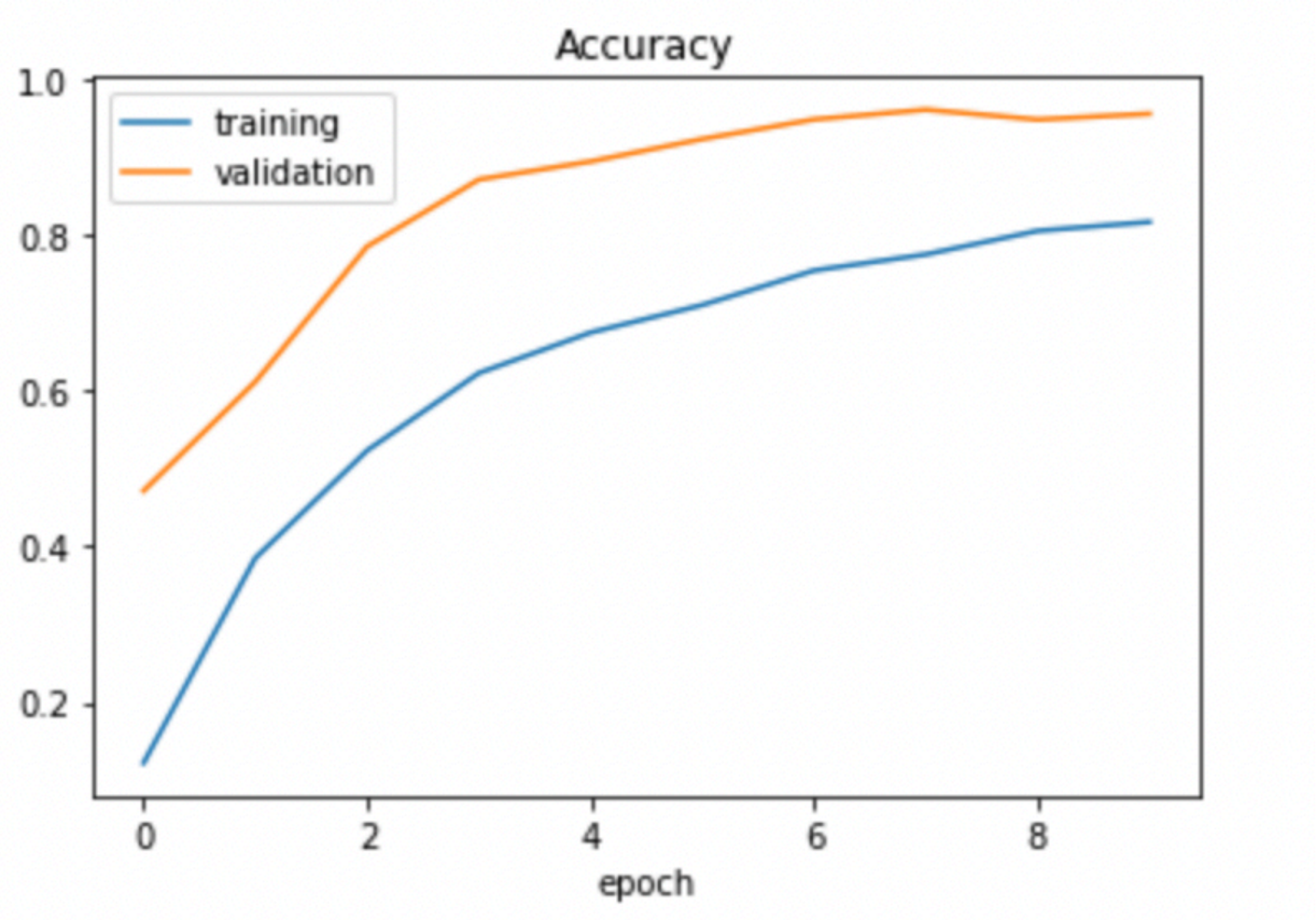
Model Summary

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 28, 28, 60)        1560

 conv2d_1 (Conv2D)           (None, 24, 24, 60)        90060

 max_pooling2d (MaxPooling2D  (None, 12, 12, 60)       0
 )

 conv2d_2 (Conv2D)           (None, 10, 10, 30)        16230

 conv2d_3 (Conv2D)           (None, 8, 8, 30)          8130

 max_pooling2d_1 (MaxPooling  (None, 4, 4, 30)         0
 2D)

 dropout (Dropout)           (None, 4, 4, 30)          0

 flatten (Flatten)           (None, 480)               0

 dense (Dense)               (None, 500)               240500

 dropout_1 (Dropout)         (None, 500)               0

 dense_1 (Dense)             (None, 43)                21543

=================================================================
Total params: 378,023
Trainable params: 378,023
Non-trainable params: 0
_________________________________________________________________
```

Model Fitting & Execution

```
Epoch 1/10
200/200 [==============================] - 15s 72ms/step - loss: 3.3031 - accuracy: 0.1226 - val_loss: 2.0919 - val_accuracy: 0.4717
Epoch 2/10
200/200 [==============================] - 14s 72ms/step - loss: 2.1384 - accuracy: 0.3855 - val_loss: 1.2403 - val_accuracy: 0.6116
Epoch 3/10
200/200 [==============================] - 15s 73ms/step - loss: 1.5678 - accuracy: 0.5234 - val_loss: 0.6830 - val_accuracy: 0.7850
Epoch 4/10
200/200 [==============================] - 15s 73ms/step - loss: 1.2321 - accuracy: 0.6227 - val_loss: 0.4409 - val_accuracy: 0.8710
Epoch 5/10
200/200 [==============================] - 15s 74ms/step - loss: 1.0453 - accuracy: 0.6747 - val_loss: 0.3572 - val_accuracy: 0.8941
Epoch 6/10
200/200 [==============================] - 15s 75ms/step - loss: 0.9074 - accuracy: 0.7100 - val_loss: 0.2369 - val_accuracy: 0.9229
Epoch 7/10
200/200 [==============================] - 15s 76ms/step - loss: 0.7977 - accuracy: 0.7543 - val_loss: 0.1972 - val_accuracy: 0.9481
Epoch 8/10
200/200 [==============================] - 15s 77ms/step - loss: 0.7109 - accuracy: 0.7749 - val_loss: 0.1624 - val_accuracy: 0.9605
Epoch 9/10
200/200 [==============================] - 15s 77ms/step - loss: 0.6219 - accuracy: 0.8051 - val_loss: 0.1504 - val_accuracy: 0.9478
Epoch 10/10
200/200 [==============================] - 16s 79ms/step - loss: 0.5829 - accuracy: 0.8165 - val_loss: 0.1500 - val_accuracy: 0.9553
```

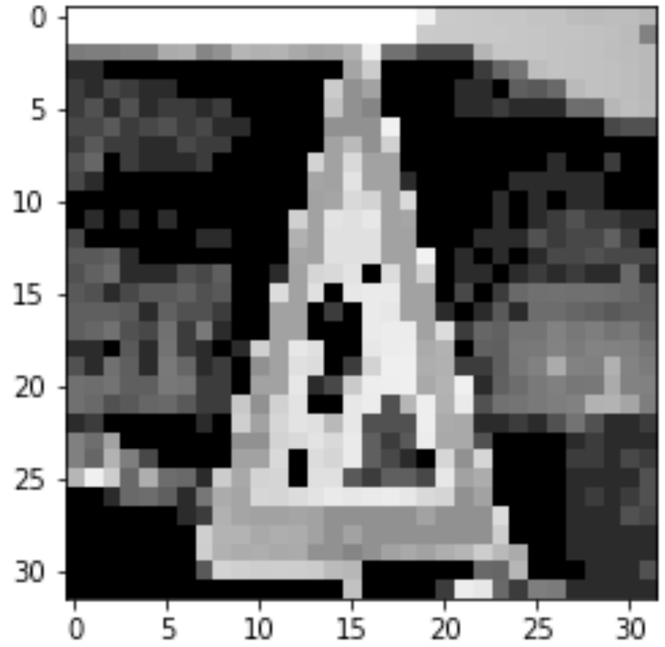
MODEL EVALUATION



Test Accuracy

0.9242280125617981

TESTING




```
#Test image
print("predicted sign: "+ str(np.argmax(model.predict(img), axis =-1 )))
```

```
predicted sign: [25]
```

25_Road work

POTENTIAL BUSINESS CHALLENGES

- Dependencies on AWS, Azure, and other cloud services
- Cost of implementation
- Weather to build in the model in house or outsource
- Integration and Adoption
- Maintenance of the model (If done in house)

THE END