EEE4114F

Machine Learning Project

Hand-drawn Symbol/Shape Classification

Department of Electrical Engineering

Written by

Crystal Jaftha [JFTCRY001]

Abu Bakr Salie [SLXABU002]

21 May 2022

1. Introduction

1.1 Subject and Motivation

This project aims to create a system using machine learning techniques that can accurately predict what is being drawn when given images. There are several applications for this development, which can range from identifying user inputs in digital interfaces to assisting with computer vision tasks.

1.2 Background

The report was created in order to show how the task of identifying images was achieved and how it can be improved upon in future. The implementation used in this report will be used to recognise shapes in images, which may be implemented on video or live footage.

1.3 Objectives

The objective of the report is to:

- Apply a machine learning algorithm to correctly identify images
- Explain how the algorithm was achieved
- Present the use of this implementation and explain how it may be implemented from images to video

1.4 Limitations and Scope

The report used the YOLOv5 object detection algorithm and will therefore only be looking at its use case. The use of YOLOv5 allows for training on images and implementation on video, which is desirable.

The problem was simplified by limiting the number of classes and using simple shapes to focus on creating a model that is as efficient and accurate as possible. The report will only identify 5 shapes, that being a square, circle, triangle, hexagon and a star. This is to limit the amount of resources required in order to process the data as just learning these shapes had taken several hours to learn. Furthermore, only images were used for training, validation and testing but how it will be transferable to video will be explained later.

1.5 Plan of Development

The report will first show the method used to collect data and train the model. It will then go on to show the results of this implementation and finally explain what the results mean and how they can be continued and improved.

2. Methodology

2.1. Data Collection and Preprocessing

The data was collected by sketching shapes on MS Paint and downloading samples from Roboflow. Five simple shapes were used for classification, and these included a circle, triangle, square, star, and hexagon. Each shape was sketched once and augmented using a Python algorithm to create more data points. A sufficient number of samples for each class was used in order for the model to be trained effectively. The images were converted into JPEG format. The images were annotated by using Makesense.ai to create bounding boxes and labels.

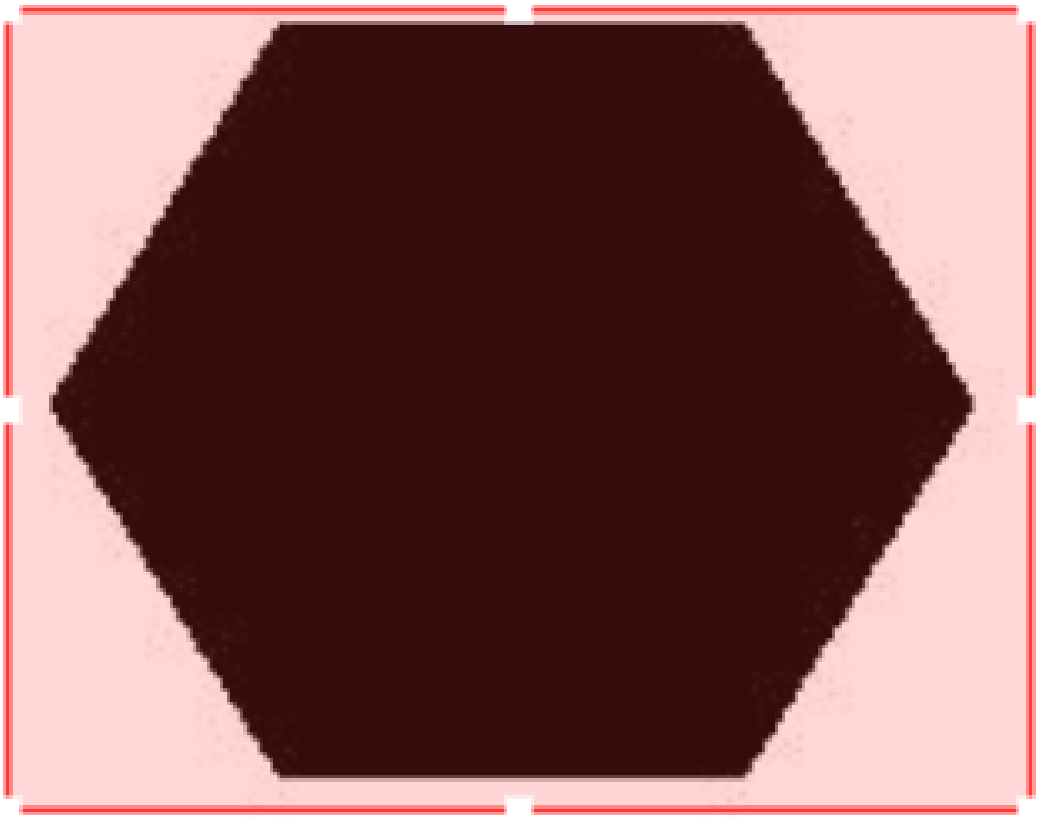

Figure 1: Annotated hexagon image

The dataset was split into training, validation, and testing sets to evaluate the performance of the model. The following table shows the number of images per dataset.

Dataset	Number of Images
Training	462
Validation	125
Testing	70

Table 1: Number of images per dataset

2.2. Model Selection

YOLOv5 is an object detecting algorithm that was used for this project. This algorithm was chosen due to its flexibility, efficiency, and accuracy. YOLOv5 is also simple to implement, as it was able to use a camera as input, and it can classify in real time efficiently. The architecture of YOLO allows object detection to be treated as a single regression problem where images are looked at once (YOLO-you only look once) to predict the objects that appear [1].

2.3. Model Training

The training and validation datasets were used to train the model. During the training process, the model learned to detect the shapes by optimising its parameters according to the given labels of the training dataset. The loss function was also optimised and the hyperparameters were altered where necessary. This process was repeated for a total of 100 epochs. The performance of the model was measured using metrics such as the F1 score, precision, and recall. A confusion matrix was used to display the performance of the model. These metrics were generated on graphs, which are shown later in the results section.

2.4. Model Evaluation and Testing

A model evaluation was done to assess how well the model performs on unseen data and to provide an overall assessment of how accurate and effective the model actually is. The evaluation was conducted using the testing dataset, which was not used during the model training. As well as the images from Roboflow, hand drawn shapes were also included in the test dataset.

3. Results and Discussion

3.1. Training and Validation

The results and metrics of the training and validation phases are presented below.

Losses and Metrics Graphs

The graphs below show the losses of the trained model and how well it performed against metrics such as precision, recall and Mean Average Precision (mAP). The graphs are plotted against the number of epochs (100), and the mAP, precision and recall were taken when the Intersection over Union (IoU) score was at 0.5 and 0.95.

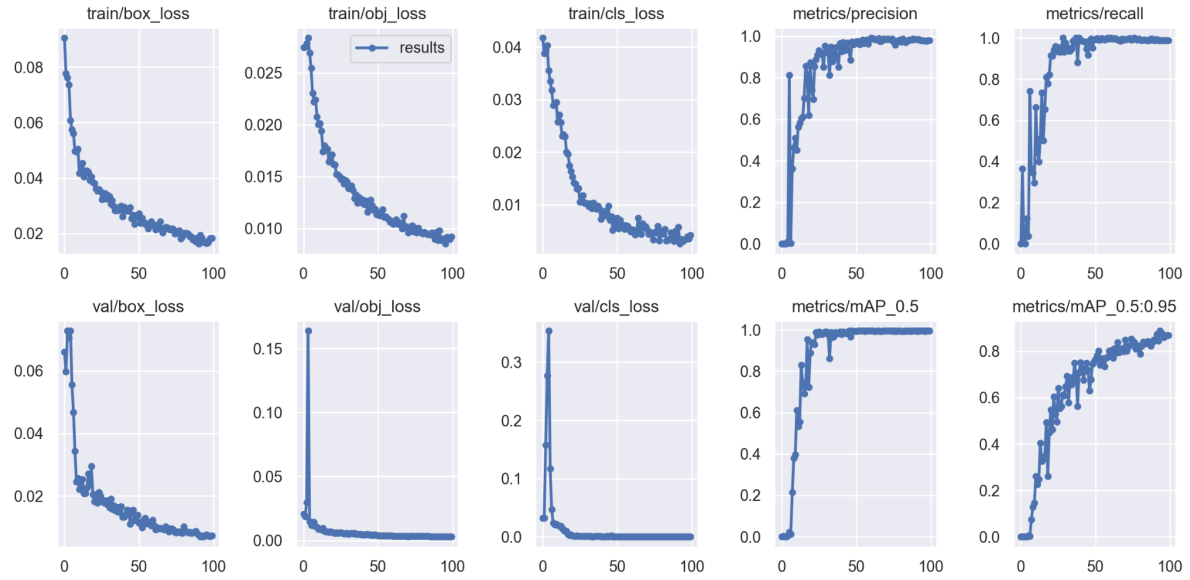

Figure 2: results showing the precision, recall and mAP of the model

From the graphs above, it can be seen that the initial losses were very high, but as the model trained, the losses started to decrease. This behaviour is typical, as the losses should be decreasing as the model learns from the data. The precision, recall and mAP graphs start off very low, but increase to values close to 1 as the model trains. These results are desirable as it indicates that the model predicts the output well.

Precision-Confidence Curve

Precision is the ratio of the number of positives that the model has predicted against the total number of true positives [2]. The figure below shows the precision confidence curve for the model.

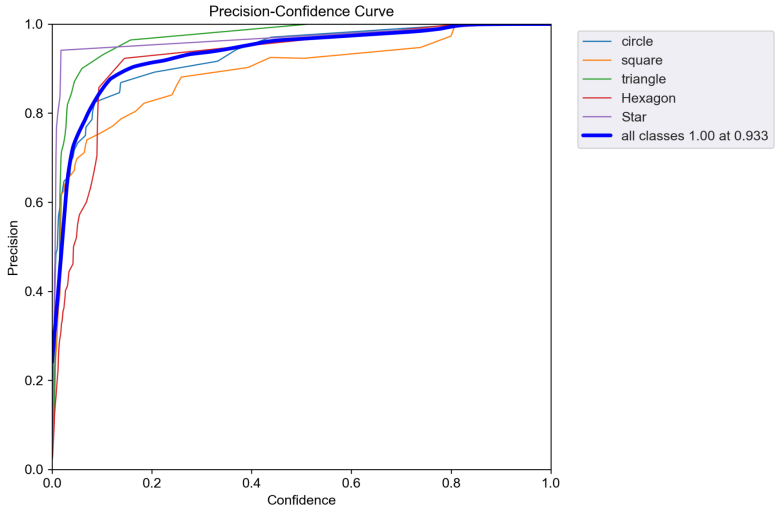

Figure 3: Precision Confidence Curve

A precision value that is close to one (or 1) is desirable, as it means that the number of predicted positives is close to the true number of positives. As seen in the precision confidence curve above, all classes have a high precision value, which gives the average precision of all the classes a value of 0.933. The model as a whole predicted the outputs very well.

Recall Confidence Curve

Recall is an indication of the model's ability to predict positive instances out of the total number of positive instances [2]. The graph below displays the recall confidence curve.

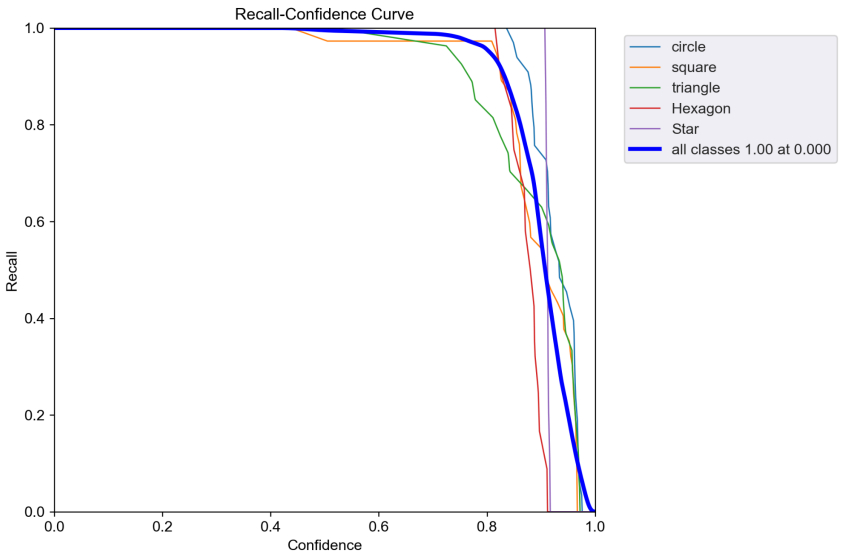

Figure 4: Recall Confidence Curve

As seen in the figure above, all classes performed well in terms of recall, as all their recall values start at 1. The square class did decrease slightly, but its value still remained high.

Precision Recall Curve

The precision recall curve combines the results of precision and recall [2].

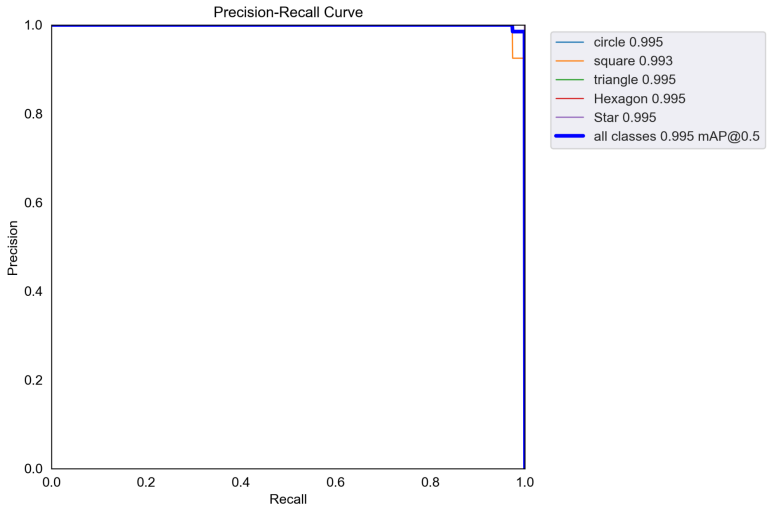

Figure 5: Precision Recall Curve

As seen in the figure above, all classes have very high precision recall values. These values are almost 1, which is desirable as it indicates that the model is well trained and is able to predict the outputs with almost 100% accuracy.

F1-Confidence Curve

The F-1 confidence curve shows the combined score of precision and recall [3].

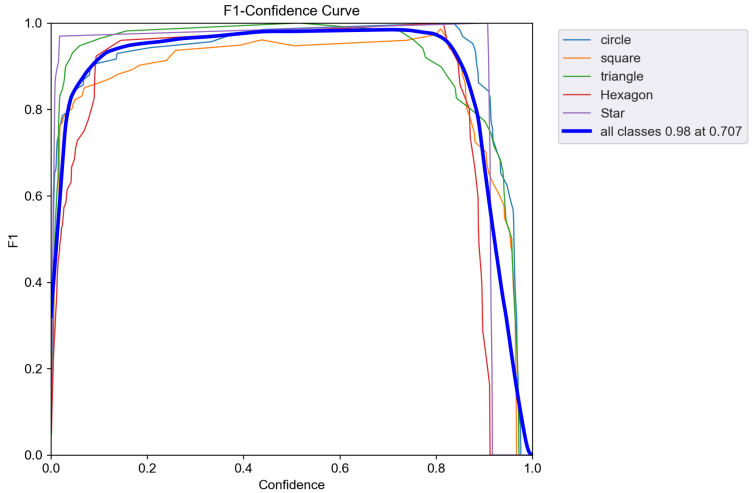

Figure 6: F1 Confidence Curve

From the F1 curve, it can be seen that 0.707 is the value that optimises precision and recall. An F1 score that is closer to 1 is desirable. As seen in the figure,this is the case for all classes, as their F1 curves are close to 1, and the average F1 score for all the classes is 0.98.

Confusion Matrix

A confusion matrix is a table that is used to assess and display the performance of a classification model. It helps with visualisation and understanding of the predictions and errors of the model and helps identify which classes are being predicted well and which aren't [3]. The following image shows the confusion matrix for the model.

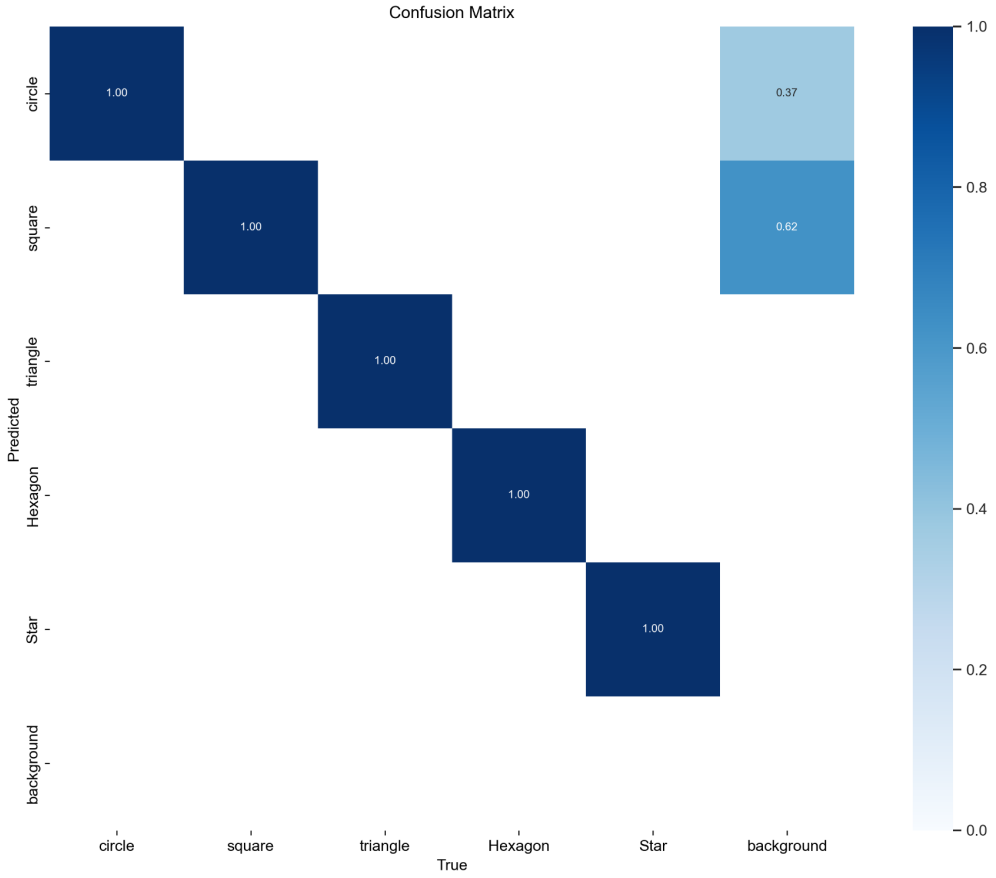

Figure 7: Confusion Matrix

As seen in the confusion matrix, the model performed well for all classes.

3.2. Testing

This section presents the results of the model training. As mentioned before, the model was trained using images that were created on Roboflow, MS Paint, as well as hand drawn images.

Hand Drawn Test Data

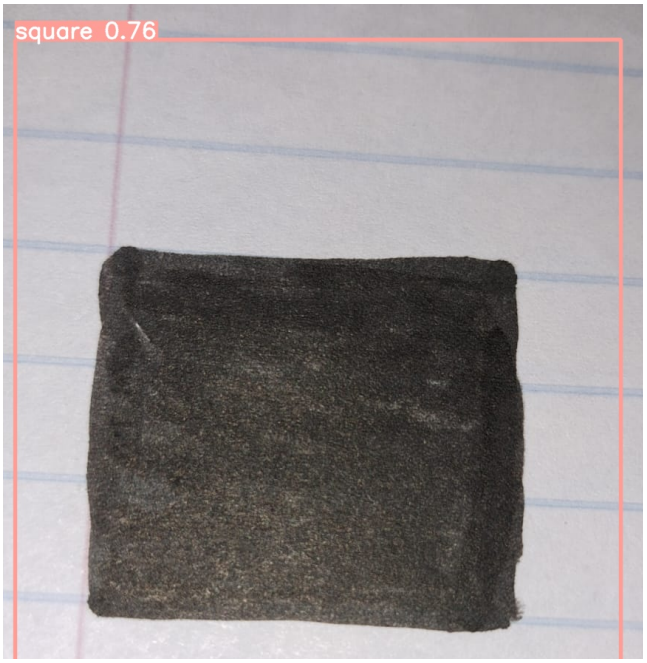

Figure 8: Hand drawn square correctly detected by model

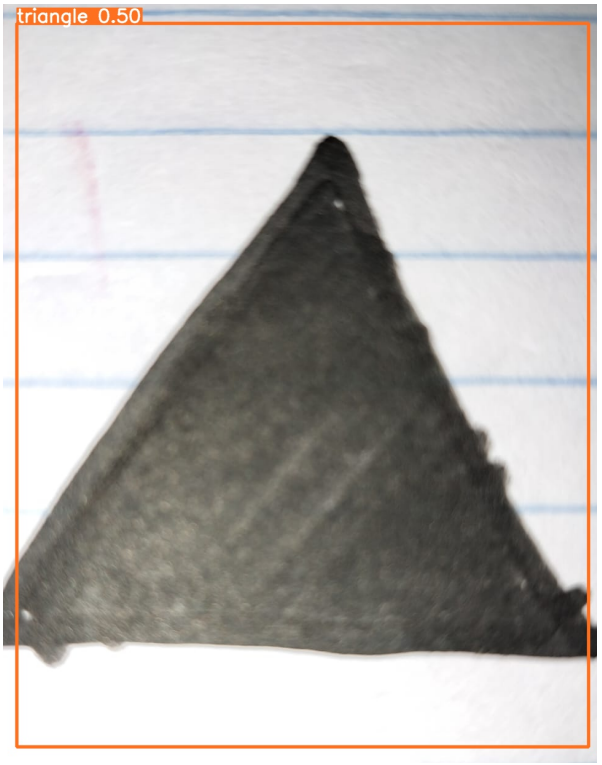

Figure 9: Hand drawn triangle correctly detected by model

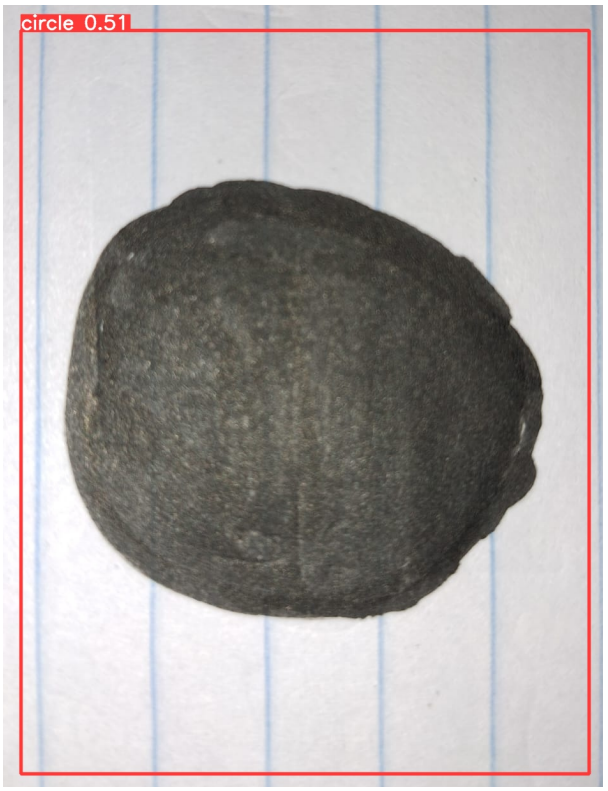

Figure 10: Hand drawn circle correctly detected by model

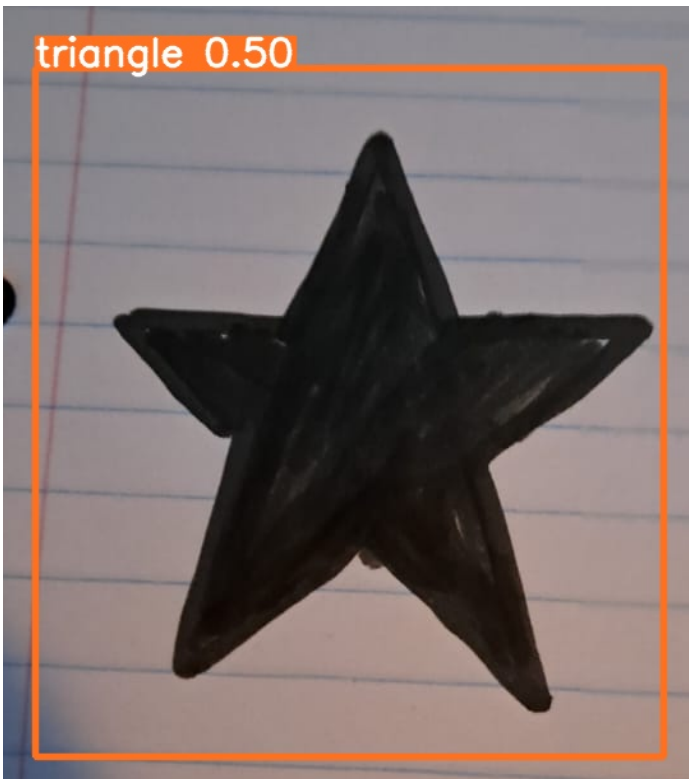

Figure 11: Hand drawn star incorrectly detected by model as triangle

Figure 12: Partial hand drawn hexagon correctly detected by model

Roboblow Test Data

Figure 13: Roboflow triangle correctly detected by model

Figure 14: Roboflow star correctly detected by model

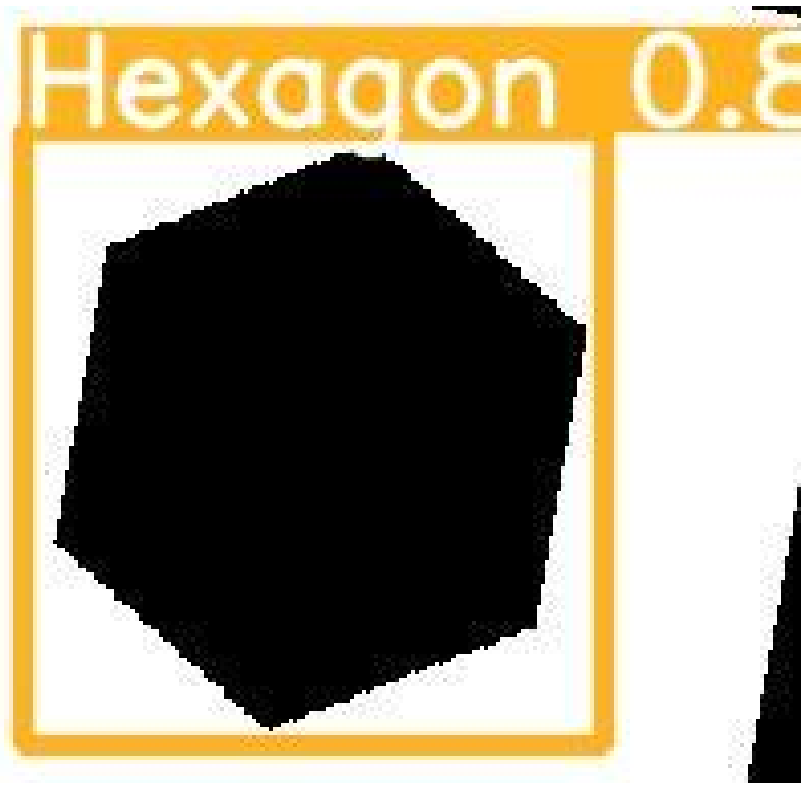

Figure 15: Roboflow hexagon correctly detected by model

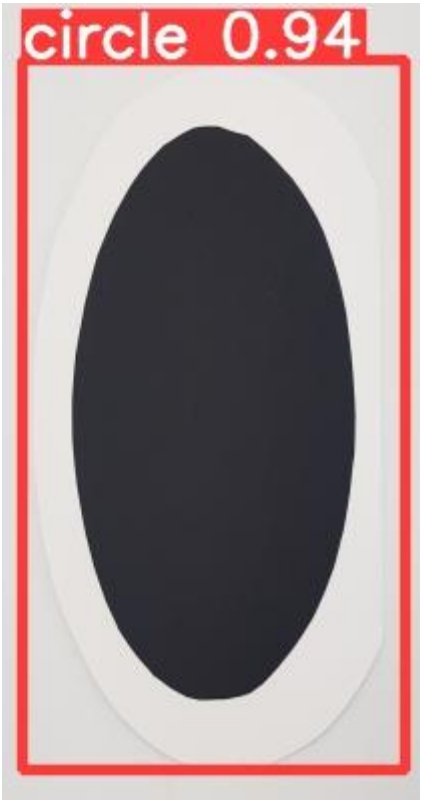

Figure 16: Roboflow circle correctly detected by model

Figure 17: Roboflow square correctly detected by model

Model Testing Discussion

The model accurately detected all 5 classes whose images were produced on Roboflow (shown in figures 13, 14, 15, 16 & 17), which was expected as it had shown high precision and recall in the metric graphs. This means that the model was trained effectively as it had generalised well to the unseen data.

The hand drawn circle, triangle, hexagon and square were all accurately detected by the model, as shown in figures 8, 9 10 & 11. Unfortunately, the star was incorrectly identified as a triangle as shown in figure 11.

Additionally, the model was able to detect partial shapes and therefore shapes can be detected while they are being drawn. This is shown in figure 12 where the hexagon is only partially drawn, but the model had correctly identified the shape as a hexagon.

4. Conclusion and Recommendations

In conclusion, a model for classifying hand-drawn shapes using the YOLOv5 algorithm was developed. Metric graphs were generated and analysed to assess the effectiveness of the model. The project was not entirely a success as it had not correctly detected the hand drawn star. Every other shape, both hand drawn and from Roboflow, was accurately detected by the model. The model therefore only needs minor improvements.

The project can be improved by increasing the training data for classes that did not perform very well i.e. some of the test data can be used in training to improve the results. Improvements can also be made by altering the hyperparameters, increasing the number of epochs and increasing the batch sizes. Additionally, hand drawn shapes could have been added to the training data to increase the chance of the model generalising well to the unseen hand drawn test data. Lastly, manually adding and labelling data is not good practice but as a tradeoff, a more dynamic model can be created that can be used for image and video transference.

5. Reference List

[1] J. Solawetz, “What is YOLOv5? A Guide for Beginners.,” Roboflow, 29 June 2020. [Online]. Available: https://blog.roboflow.com/yolov5-improvements-and-evaluation/. [Accessed 21 May 2023].

[2] Z. LT, “Precision and Recall Made Simple,” Towards Data Science, 9 November 2021. [Online]. Available: https://towardsdatascience.com/precision-and-recall-made-simple-afb5e098970f. [Accessed 21 May 2023].

[3] Z. LT, “Essential Things You Need to Know About F1-Score,” Towards Data Science, 23 November 2021. [Online]. Available: https://towardsdatascience.com/essential-things-you-need-to-know-about-f1-score-dbd973bf1a3#:~:text=The%20higher%20the%20precision%20and,1%2C%20the%20better%20the%20model.. [Accessed 21 May 2023].