Traffic Sign Classification Research Report

Christopher Heimbuch – Flatiron School Data Science

As of September 12, 2024

Section I

Executive Summary

Executive Summary

As a reminder, the dataset consists of 58 different classes of traffic signs. It was solely image data, with an excel report that outlined which each class label was.

Key Findings:

- **Many Traffic Images:**
  - 4,170 total traffic sign images curated in this data set to allow a model to learn.
- **Heavy Class Imbalance:**
  - “Watch Out For Cars” sign class had 446 images, while “Keep Left” and “Give Way” sign class only had 2 images
- **Generally Consistent Image Dimensions:**
  - The majority of images had roughly 140 pixel width and 140 pixel height. Very few images were over 300x300 dimensionality

- **Machine Learning Results:**
  - **Shotgun Approach:** Out of 6 tested classification models, the Stochastic Gradient Descent (SGD), and SVM models were selected for further testing.
  - **Hyperparameter Tuning:** SDG performed only slightly better, where SVM performed substantially worse
  - **Class Rebalance:** Instead of 58 classes, I grouped traffic images into 3 main sets: Directional, Speed limit, and warning signs.
  - **Results:** SGD model achieved an overall accuracy and F1 score of 96%.

- **Deep Learning CNN Results:**
  - **Custom Architecture:** A custom CNN model was designed, opting not to use a pretrained model.
  - **Training Performance** Model achieved 99.13% training accuracy & 99.52% validation accuracy
  - **Results:** Model tested on newly introduced testing images, achieving 93.07% accuracy

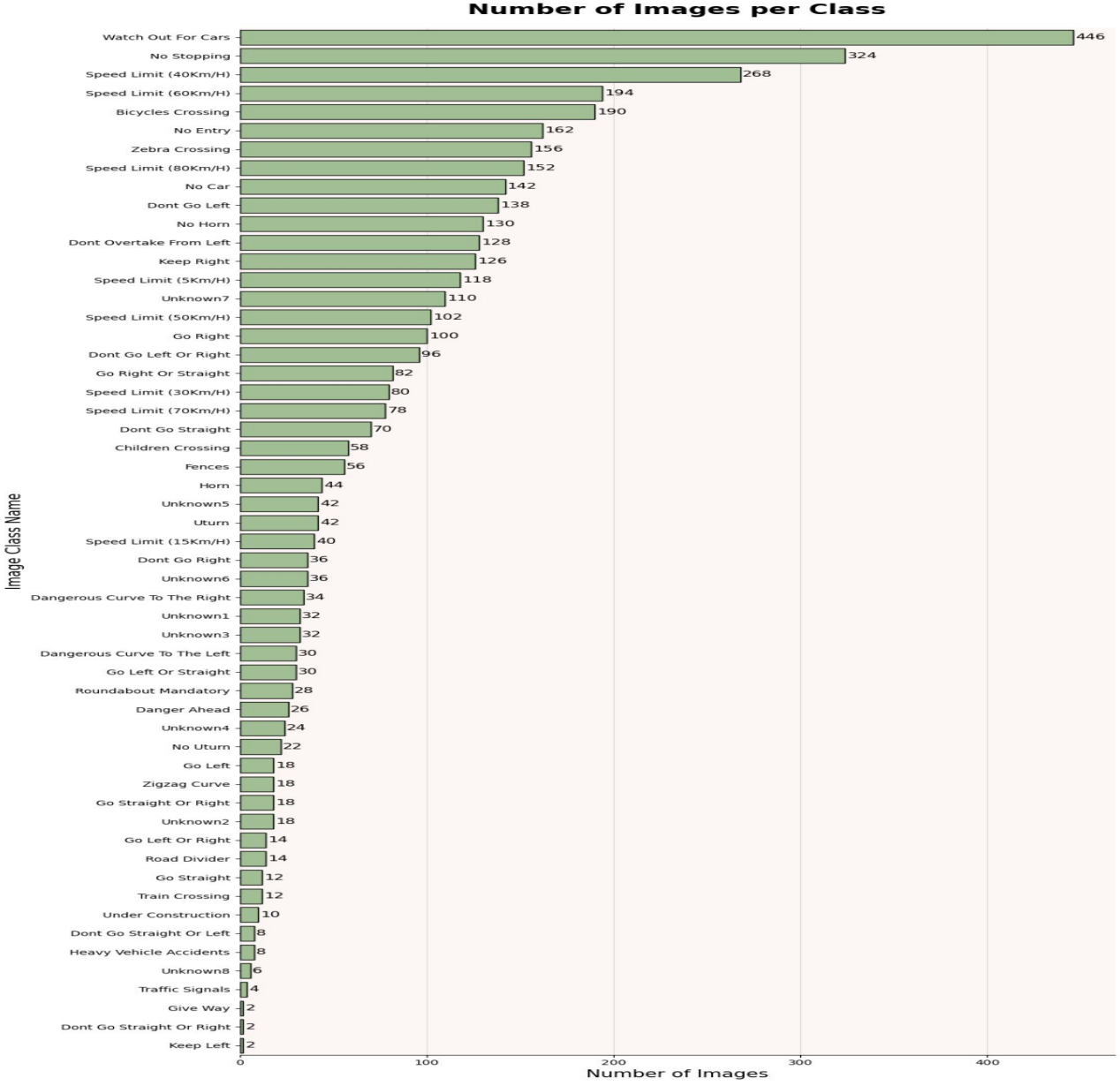
Recommendations for Companies:

With this custom pretrained traffic sign architecture, you could deploy it RIGHT NOW! This model can be used for self driving vehicles, traffic management systems, mobile apps for drivers (similar to google maps), robotics such as delivery robots, traffic law enforcement, toll collection zones, and many more!

Section II

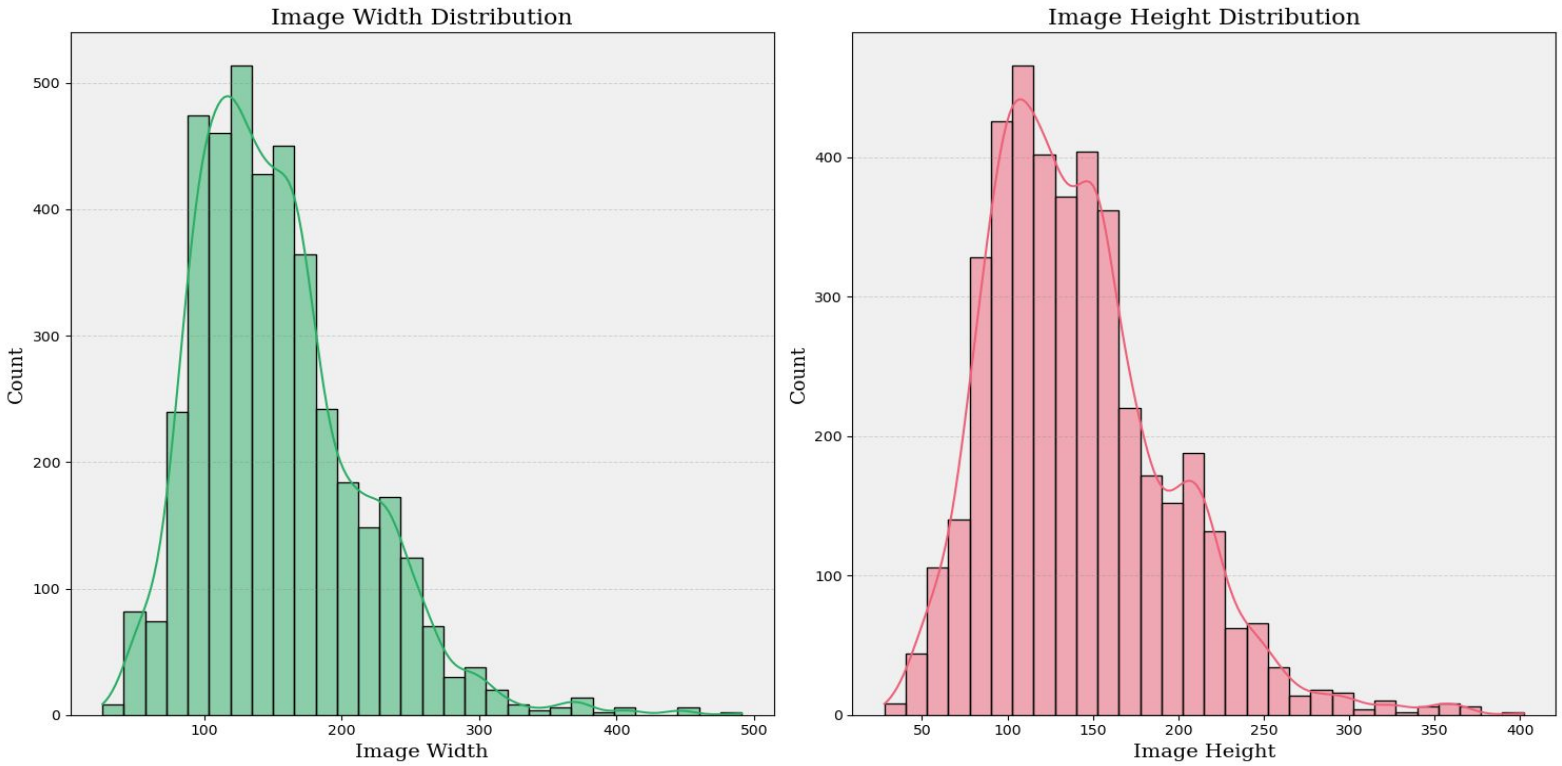
Data Visualization, Machine Learning Approach,
& Deep Learning

What is the Class Distribution?



- **Top Sign Classes:** ‘Watch Out For Cars’, ‘No Stopping’, and Speed limit signs were among the classes with highest count of signs
- **Least Sign Classes:** ‘Keep Left’, ‘Give Way’ among many others only had 2 images in their class

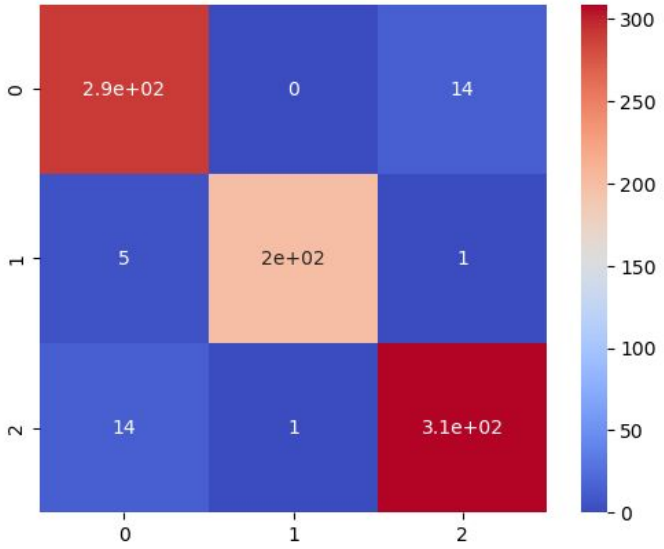
What is Image dimensionality among class distribution?



- **Image dimensionality was almost normally distributed, but more positively skewed.**
- **Most images fell between 100 - 200 pixels for width and height.**

Machine Learning Approach (1 of 2)

- **Data Preparation:** Class rebalance from 58 to 3 main classes (Speed Limit signs, Directional, Warning)
- **Model Selection:** I used a shotgun approach, which I used 6 classifiers: Logistic Regression, ADA Boost Classifier, SVC, Gradient Boosting, SGD, and Random Forest.
- **Top Performers:** The models I moved forward with were SVC and SGD. After hyper parameter tuning, SGD performed only slightly better and SVC performed substantially worse.



	precision	recall	f1-score	support
0	0.94	0.95	0.95	304
1	1.00	0.97	0.98	206
2	0.95	0.95	0.95	324
accuracy			0.96	834
macro avg	0.96	0.96	0.96	834
weighted avg	0.96	0.96	0.96	834

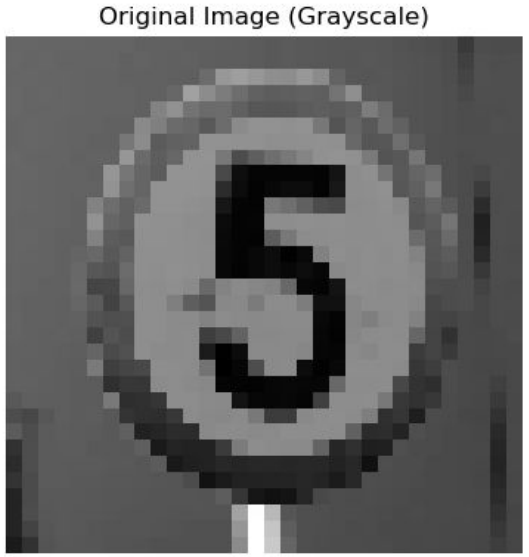
'Accuracy Score: 0.96'

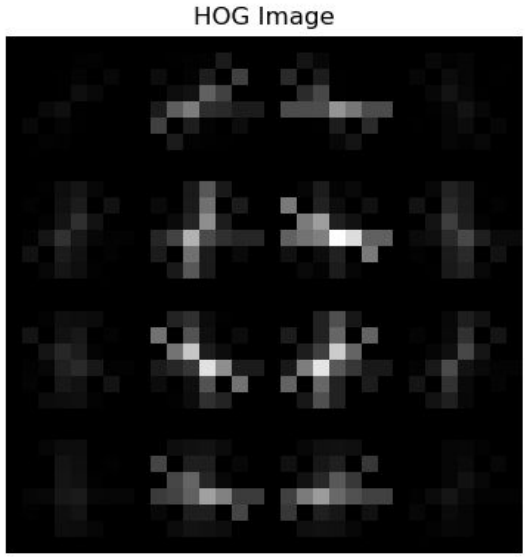
*Results shown for SGD classifier

Machine Learning Approach (2 of 2)

Histogram of Oriented Gradients

- **HOG Application:** I applied the HOG method, which is a feature descriptor user in computer vision and image processing for object detection. I attempted to see if there was any improvement in accuracy.
- **Impact on Accuracy:** Some models performed worse, while only a few performed slightly better
- **Model Performance:** Compared to my other approach, SGD performed worse, with a 91% accuracy!





Deep Learning Approach (CNN)

CNN Deep Learning Approach

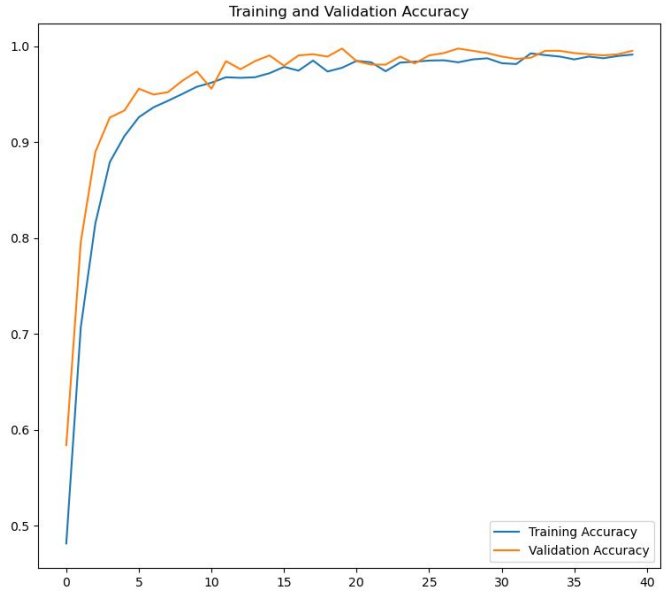
- **CNN Approach:** I designed a Convolutional Neural Network from scratch to learn on my image data
- **Image Preprocessing:** I converted all images to 32x32x3, kept them in 2D format, and normalized them by dividing all pixel dimensions by 255 to get all pixel data in a format so it their values were between 0 - 1.
- **Model Training:** The CNN model was trained over 40 epochs

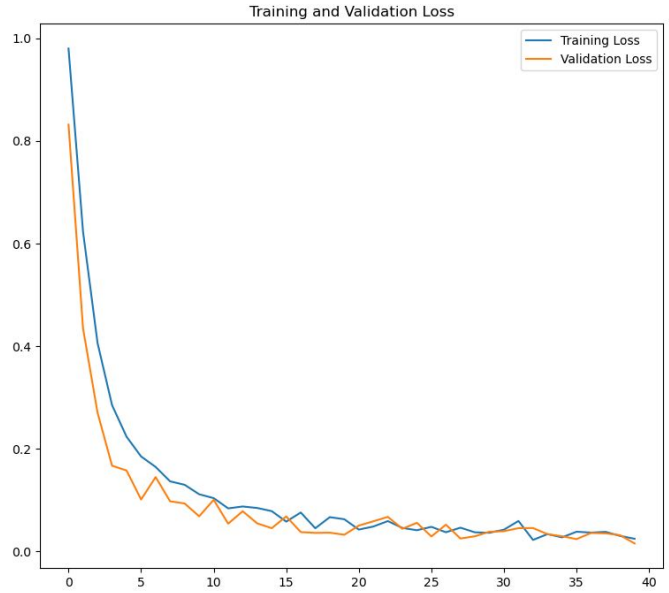
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 32, 32, 50)        1400

 conv2d_1 (Conv2D)           (None, 32, 32, 75)        33825

 max_pooling2d (MaxPooling2D  (None, 16, 16, 75)       0
 )

 dropout (Dropout)           (None, 16, 16, 75)        0

 conv2d_2 (Conv2D)           (None, 16, 16, 125)       84500

 max_pooling2d_1 (MaxPooling  (None, 8, 8, 125)        0
 2D)

 dropout_1 (Dropout)         (None, 8, 8, 125)         0

 flatten (Flatten)           (None, 8000)              0

 dense (Dense)               (None, 500)               4000500

 dropout_2 (Dropout)         (None, 500)               0
...
Total params: 4,246,228
Trainable params: 4,246,228
Non-trainable params: 0
```

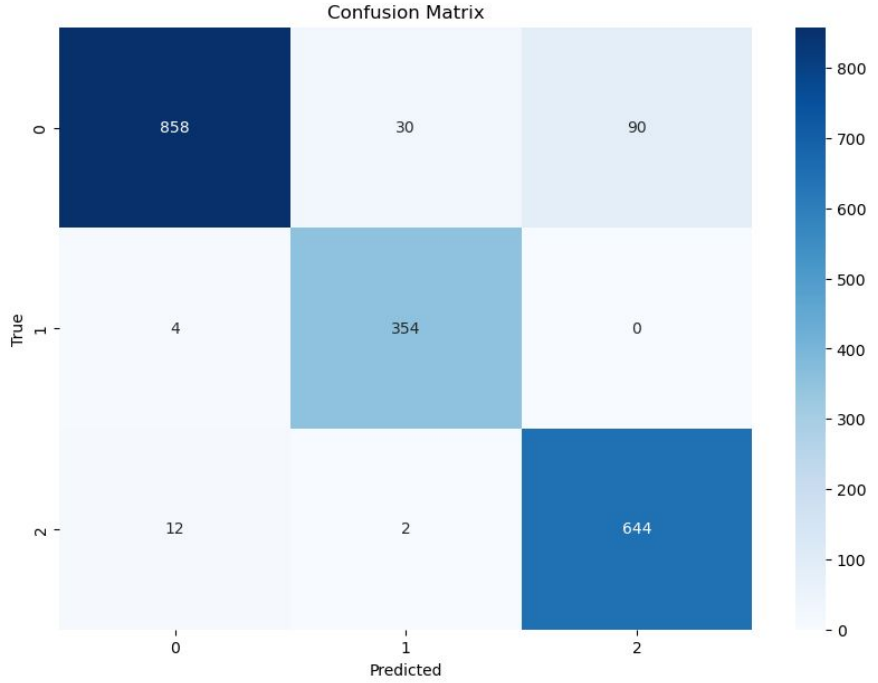
CNN Results

CNN Deep Learning Approach

- **Training Performance:** The CNN achieved a best accuracy of 99.13%, with a validation accuracy of 99.52%
- **Testing Performance:** The confusion matrix to the right was test images introduced to the model in which it has never seen, and it scored an accuracy of 93.08%.

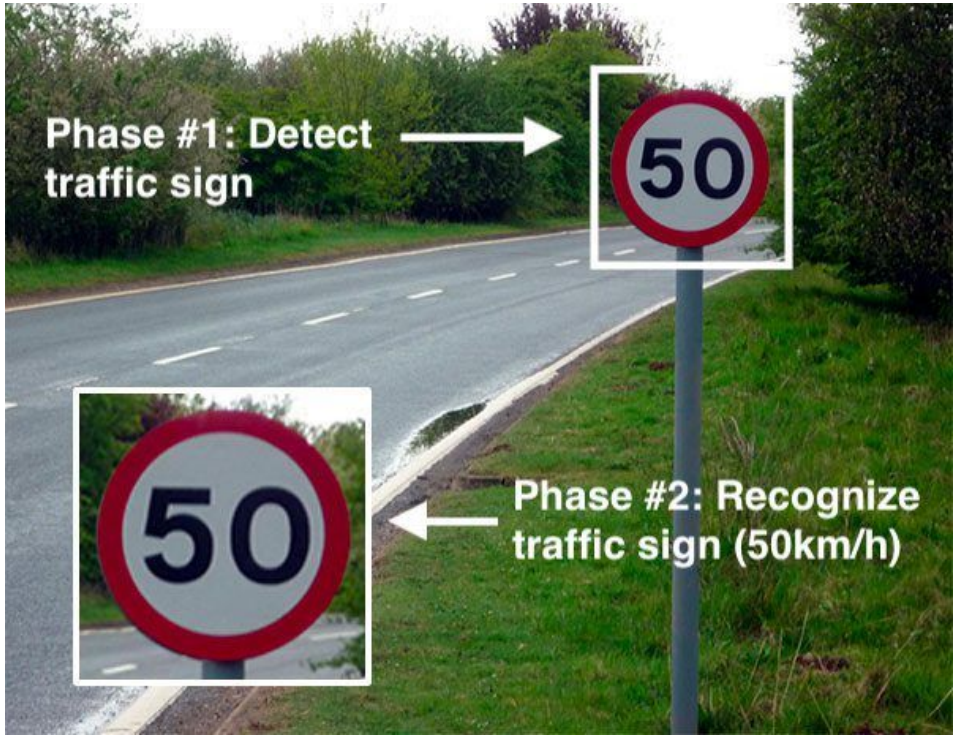






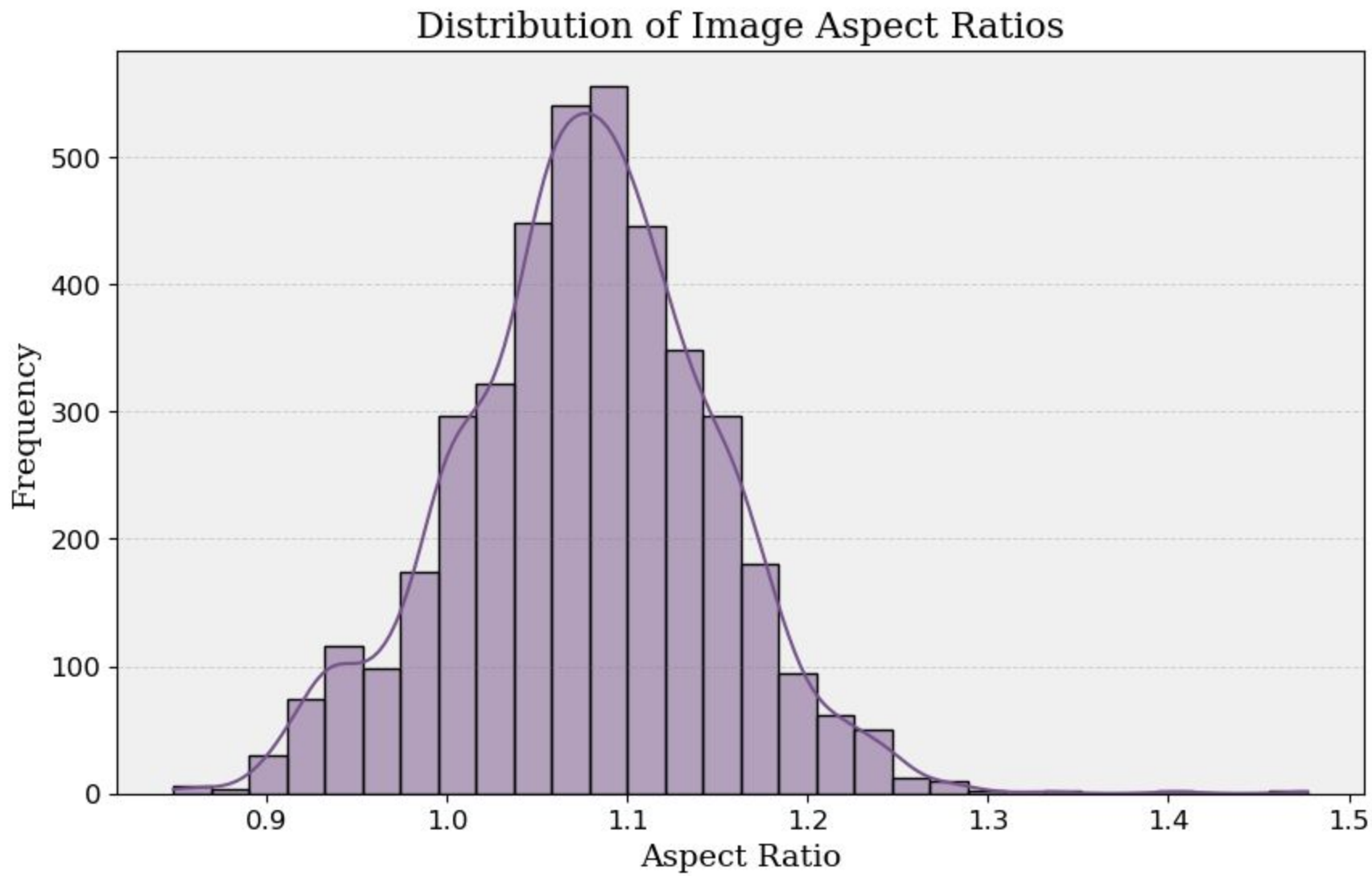
Recommendation

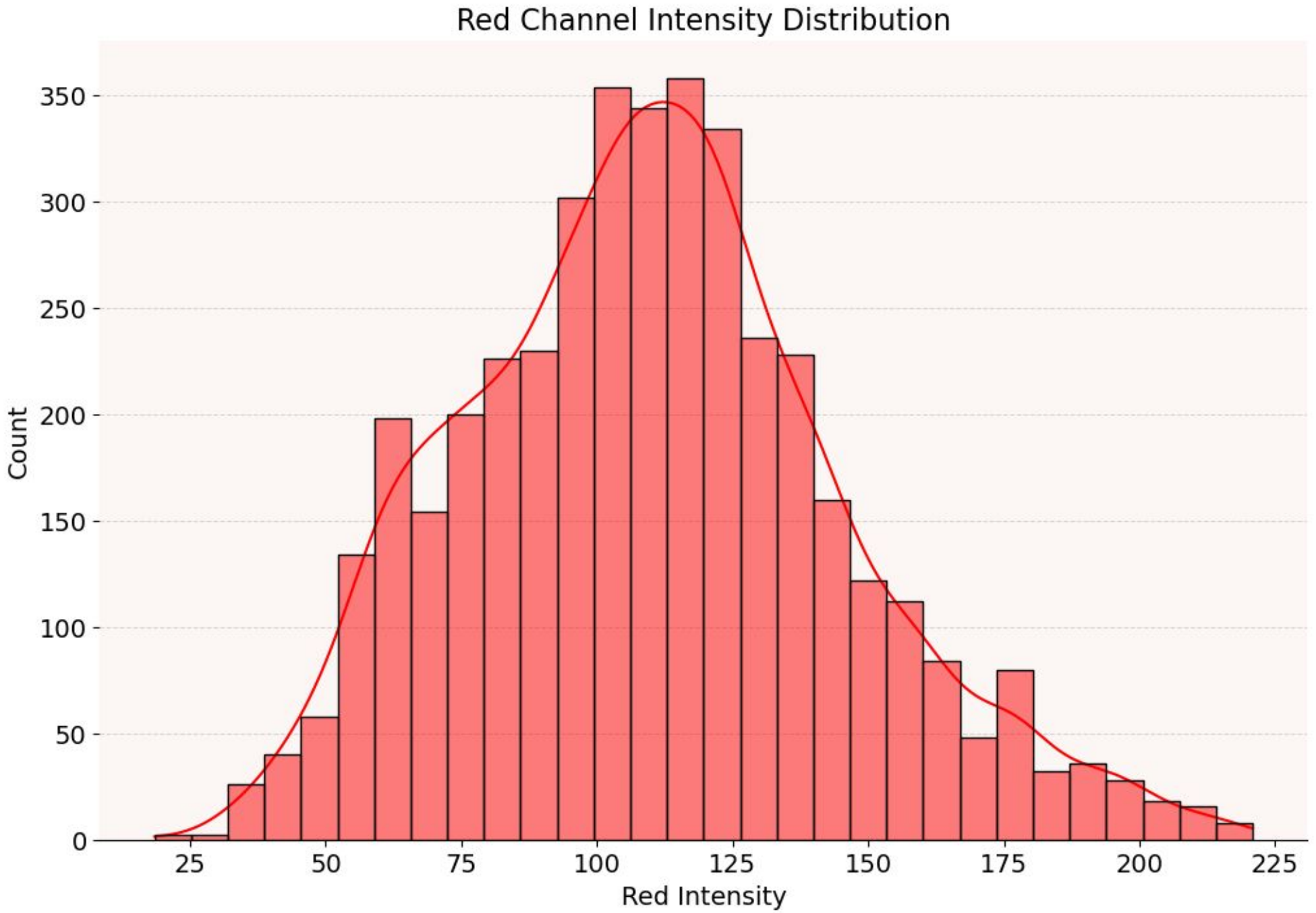
- This custom CNN pre trained model is ready to be deployed. It can be used in a variety of scenarios. It can be used for self driving vehicles, traffic management systems, mobile apps for drivers (similar to google maps), robotics such as delivery robots, traffic law enforcement, toll collection zones, and many more! For companies who find themselves needing to enhance their applications, or for law enforcement that need to advanced systems that can detect numerous objects that include traffic signs, this can prove to be incredibly useful for them.

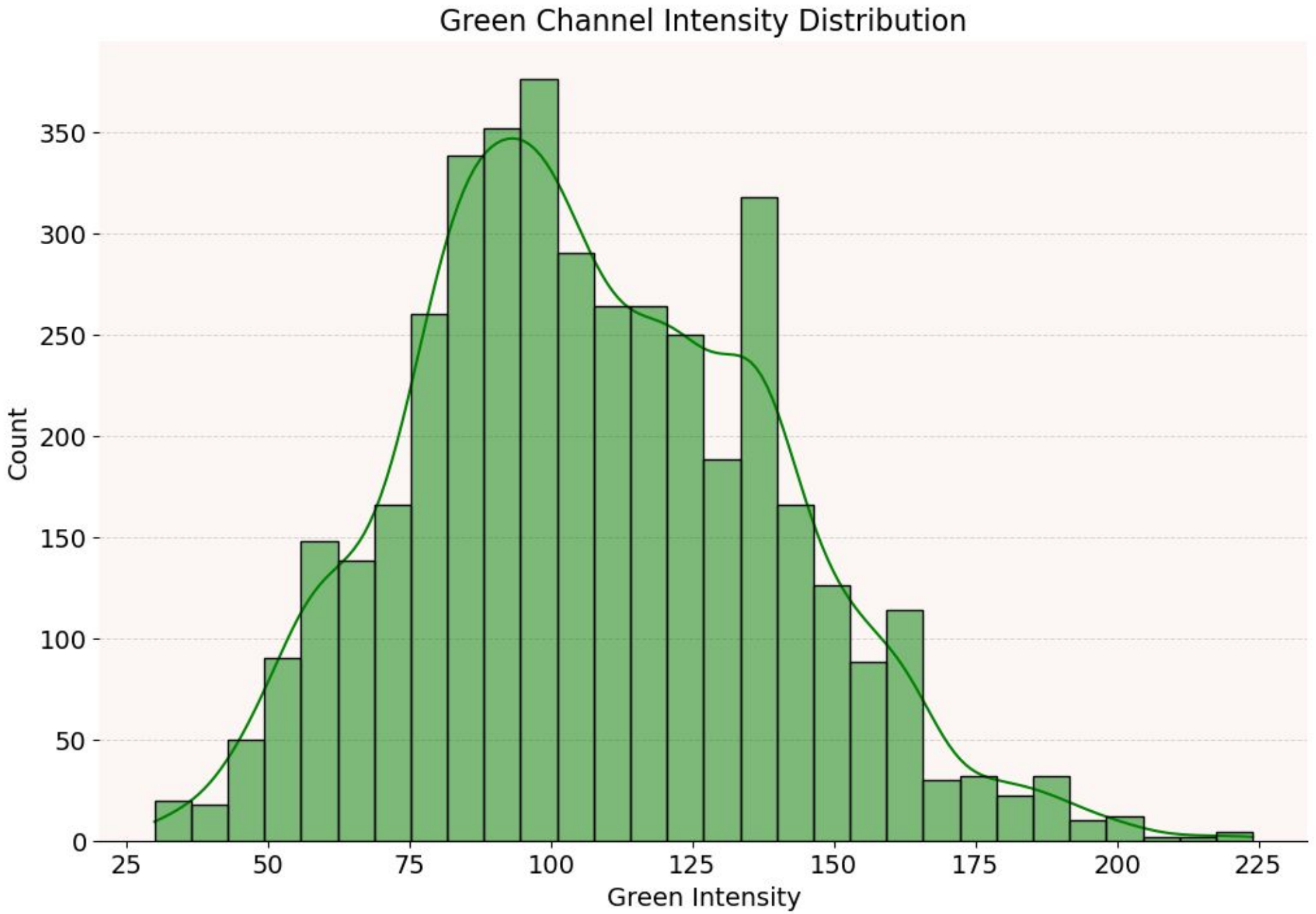


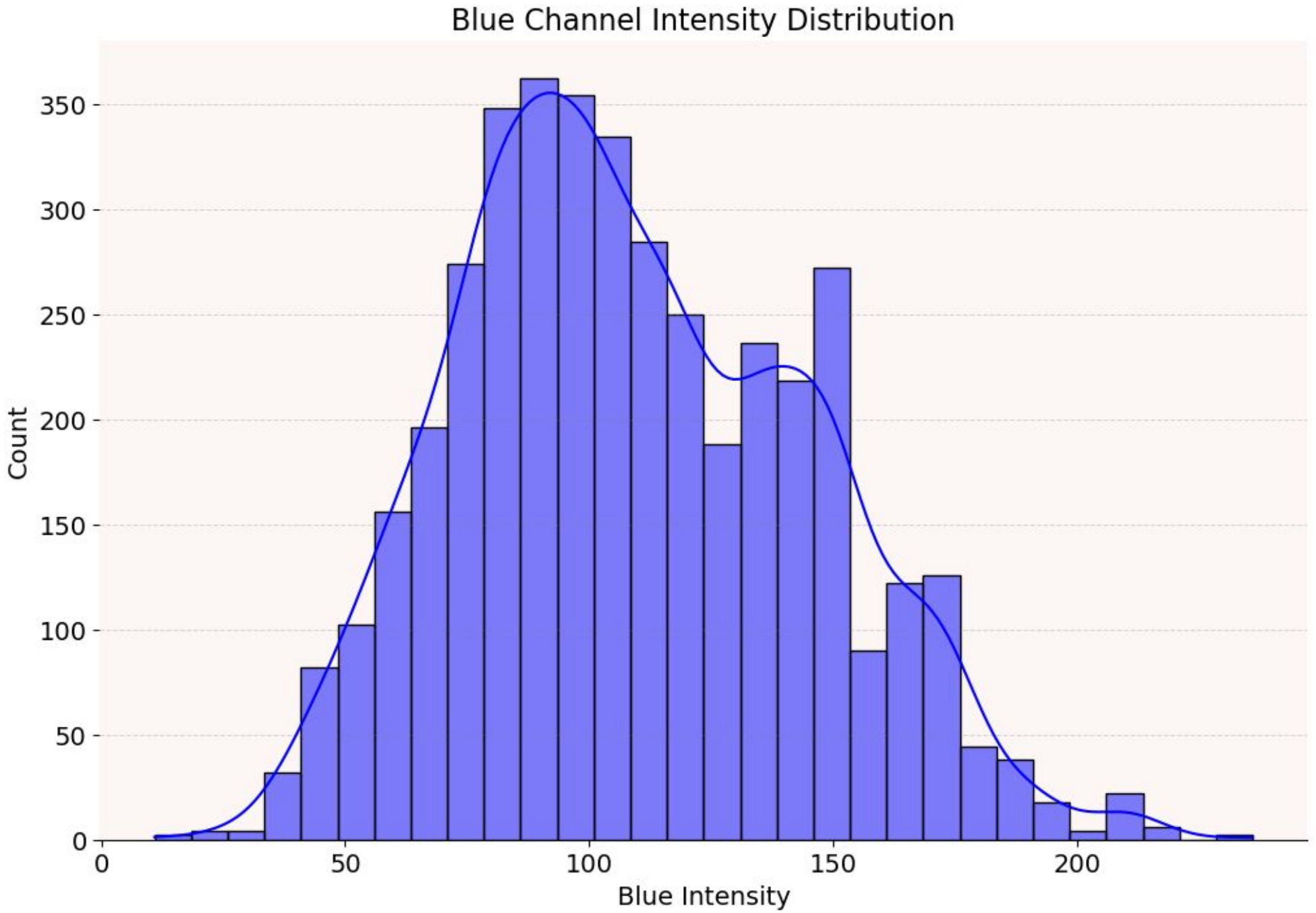
Section III

Appendix


Distribution of Image Aspect Ratios
Frequency
0
100
200
300
400
500
0.9
1.0
1.1
1.2
1.3
1.4
1.5
Aspect Ratio


Red Channel Intensity Distribution
Count
Red Intensity
0
50
100
150
200
250
300
350
25
50
75
100
125
150
175
200
225


Green Channel Intensity Distribution
Count
0
50
100
150
200
250
300
350
25
50
75
100
125
150
175
200
225
Green Intensity


Blue Channel Intensity Distribution
Count
Blue Intensity
0
50
100
150
200
0
50
100
150
200
250
300
350