In [3]:
```
#Import required libraries
import cv2
import time
import numpy as np
import pandas as pd
import tensorflow as tf
import matplotlib.pyplot as plt
import os
from os import listdir
from os.path import isfile, join
```

In [21]:
```
#Read the Train_data_label Excel file
train_data = pd.read_excel("Dropbox/Coding/AI and ML Bootcamp/Course 4 Deep Learning w
```

In [32]:
```
#Read train folder images and resize them to 30x30
#Create a NumPy array from resized images
base_path = "Dropbox/Coding/AI and ML Bootcamp/Course 4 Deep Learning with Keras and T
def read_and_resize(base_path, image_paths):
    train_resized = []
    for path in image_paths:
        full_path = base_path + "/" + path
        image = cv2.imread(full_path)
        image = cv2.resize(image, (30, 30))
        train_resized.append(image)
    return np.array(train_resized)
train_resized = read_and_resize(base_path, train_data['Path'].values)
```

In [33]:
```
#Check number and dimensions of images
print(train_resized.shape)
```

```
(39209, 30, 30, 3)
```

In [34]:
```
#Convert RGB images to grayscale
grayscale_img = [cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) for img in train_resized]
#Check length of arrays to make sure they match
print(len(train_resized))
print(len(grayscale_img))
```

```
39209
39209
```

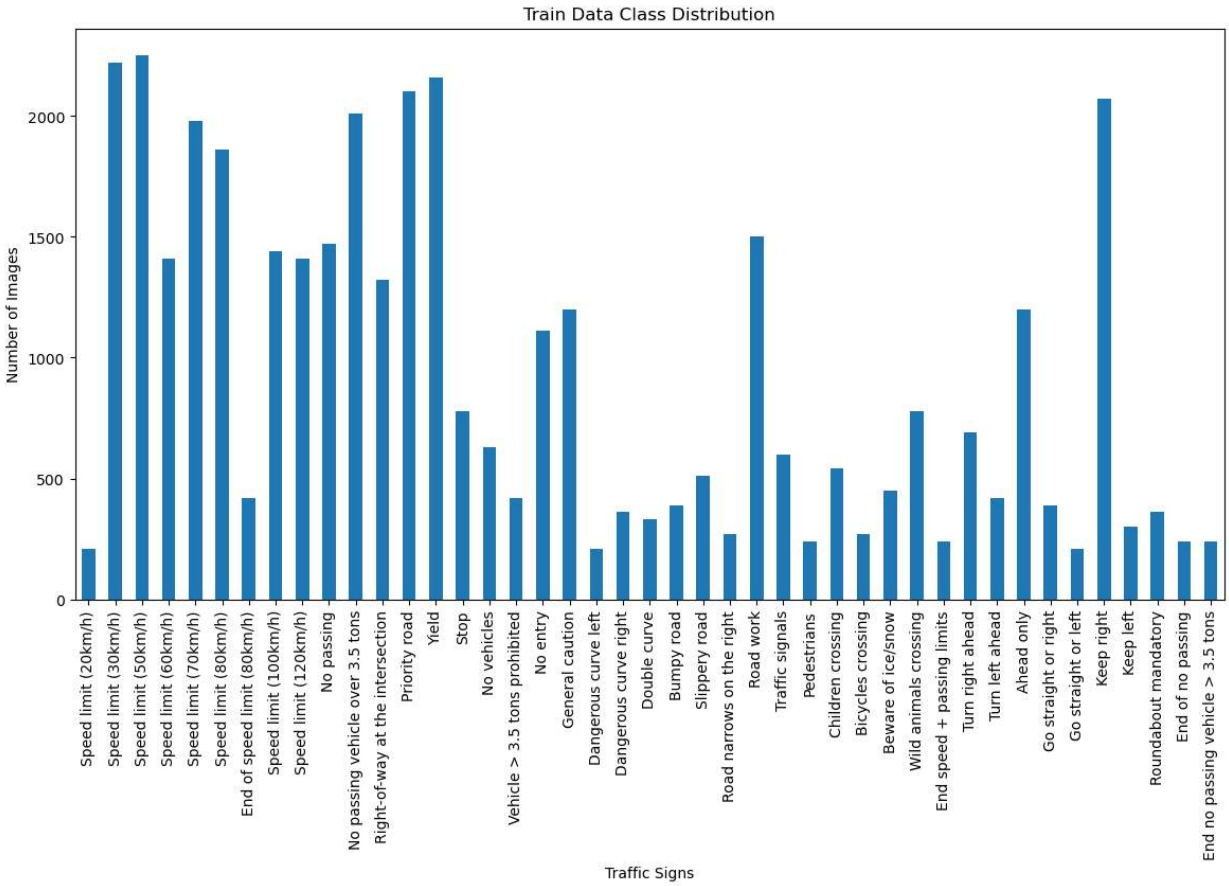
In [35]:
```
#Save color image and grayscale image arrays in new columns in train_data
train_data['color_img_arr'] = [img for img in train_resized]
train_data['grayscale_img_arr'] = [img for img in grayscale_img]
```

In [36]:
```
#Import 'test_data' for bar chart
test_data = pd.read_excel('Dropbox/Coding/AI and ML Bootcamp/Course 4 Deep Learning wi
```

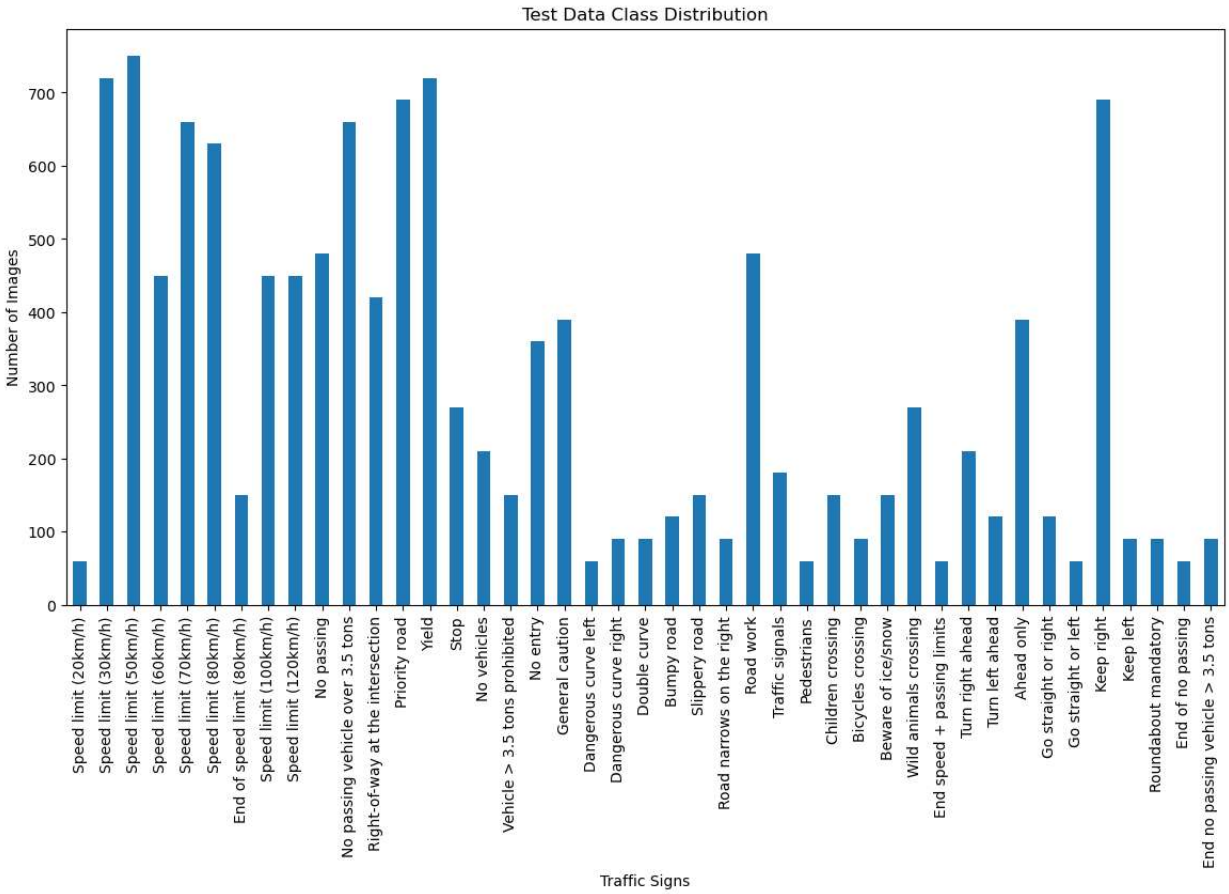
In [38]:
```
#Plot bar chart of classes from train and test data with actual class names
#Get value counts of classes in datasets and organize them by index
train_class_count = train_data['ClassId'].value_counts().sort_index()
test_class_count = test_data['ClassId'].value_counts().sort_index()
#Define class names in index order
class_names = ["Speed limit (20km/h)", "Speed limit (30km/h)", "Speed limit (50km/h)",
               "End of speed limit (80km/h)", "Speed limit (100km/h)", "Speed limit (1
               "Right-of-way at the intersection", "Priority road", "Yield", "Stop", "
               "Dangerous curve left", "Dangerous curve right", "Double curve", "Bumpy
```

```
                "Pedestrians", "Children crossing", "Bicycles crossing", "Beware of ice
                "Turn left ahead", "Ahead only", "Go straight or right", "Go straight o
                "End no passing vehicle > 3.5 tons"]
```

In [43]:
```
#Plot classes from train data
plt.figure(figsize=(14,7))
train_class_count.plot(kind='bar')
plt.xticks(ticks=range(len(class_names)), labels=class_names, rotation=90)
plt.title('Train Data Class Distribution')
plt.xlabel('Traffic Signs')
plt.ylabel('Number of Images')
plt.show()
```



In [44]:
```
#Plot classes from test data
plt.figure(figsize=(14,7))
test_class_count.plot(kind='bar')
plt.xticks(ticks=range(len(class_names)), labels=class_names, rotation=90)
plt.title('Test Data Class Distribution')
plt.xlabel('Traffic Signs')
plt.ylabel('Number of Images')
plt.show()
```



In [47]:
```
#Prepare the model's training and testing data and convert to appropriate format and s
#Normalize training images
train_color = train_resized.astype('float32') / 255
grayscale_img_array = np.array(grayscale_img)
train_grayscale = grayscale_img_array.astype('float32') / 255

#Reshape training grayscale images to have a depth dimension (make grayscale images 3-
train_grayscale = train_grayscale.reshape(-1, 30, 30, 1)
```

In [48]:
```
#One-hot encode the training dataset labels
from tensorflow.keras.utils import to_categorical
train_labels = train_data['ClassId'].values
train_labels_encoded = to_categorical(train_labels, num_classes=43)
```

In [49]:
```
#Split color and grayscale images into training and validation sets
from sklearn.model_selection import train_test_split
X_train_color, X_val_color, y_train_color, y_val_color = train_test_split(train_color,
X_train_gray, X_val_gray, y_train_gray, y_val_gray = train_test_split(train_grayscale,
```

In [50]:
```
#Read test folder images and resize them to 30x30
#Create a NumPy array from resized images
base_path = "Dropbox/Coding/AI and ML Bootcamp/Course 4 Deep Learning with Keras and T
test_resized = read_and_resize(base_path, test_data['Path'].values)
```

In [54]:
```
#Normalize test dataset, convert to grayscale, and one-hot encode labels
test_gray = [cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) for img in test_resized]
test_gray = np.array(test_gray)
test_color = test_resized.astype('float32') / 255
test_gray = test_gray.astype('float32') / 255
```

```
test_gray = test_gray.reshape(-1, 30, 30, 1)

test_labels = test_data['ClassId'].values
test_labels_encoded = to_categorical(test_labels, num_classes=43)
```

In [55]:
```
#Create CNN model for color images
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

color_model = Sequential()

#Add convolutional, MaxPool, and dropout layers with relu activation function
color_model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(30, 30, 3)))
color_model.add(MaxPooling2D((2, 2)))
color_model.add(Dropout(0.25))

color_model.add(Conv2D(64, (3, 3), activation='relu'))
color_model.add(MaxPooling2D((2, 2)))
color_model.add(Dropout(0.25))

color_model.add(Flatten())
color_model.add(Dense(256, activation='relu'))
color_model.add(Dropout(0.5))
color_model.add(Dense(43, activation='softmax'))
```

In [56]:
```
#Create CNN model for grayscale images
grayscale_model = Sequential()

grayscale_model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(30, 30, 1)))
grayscale_model.add(MaxPooling2D((2, 2)))
grayscale_model.add(Dropout(0.25))

grayscale_model.add(Conv2D(64, (3, 3), activation='relu'))
grayscale_model.add(MaxPooling2D((2, 2)))
grayscale_model.add(Dropout(0.25))

grayscale_model.add(Flatten())
grayscale_model.add(Dense(256, activation='relu'))
grayscale_model.add(Dropout(0.5))
grayscale_model.add(Dense(43, activation='softmax'))
```

In [57]:
```
#Compile models with loss function as categorical cross-entropy and Adam optimizer
color_model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accur
grayscale_model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['a
```

In [58]:
```
#Prepare early stopping with two patience and validation loss monitoring
from tensorflow.keras.callbacks import EarlyStopping
early_stop = EarlyStopping(monitor='val_loss', patience=2)
```

In [59]:
```
#Experiment with both models for five epochs with early stopping
color_history = color_model.fit(X_train_color, y_train_color, validation_data=(X_val_c
grayscale_history = grayscale_model.fit(X_train_gray, y_train_gray, validation_data=(X
```

```
Epoch 1/5
491/491 [==============================] - 11s 21ms/step - loss: 1.6484 - accuracy:
0.5286 - val_loss: 0.4589 - val_accuracy: 0.8854
Epoch 2/5
491/491 [==============================] - 11s 23ms/step - loss: 0.4487 - accuracy:
0.8599 - val_loss: 0.1636 - val_accuracy: 0.9587
Epoch 3/5
491/491 [==============================] - 12s 25ms/step - loss: 0.2497 - accuracy:
0.9220 - val_loss: 0.0903 - val_accuracy: 0.9773
Epoch 4/5
491/491 [==============================] - 12s 25ms/step - loss: 0.1760 - accuracy:
0.9459 - val_loss: 0.0650 - val_accuracy: 0.9858
Epoch 5/5
491/491 [==============================] - 12s 25ms/step - loss: 0.1323 - accuracy:
0.9594 - val_loss: 0.0493 - val_accuracy: 0.9876
Epoch 1/5
491/491 [==============================] - 11s 20ms/step - loss: 2.0333 - accuracy:
0.4504 - val_loss: 0.6056 - val_accuracy: 0.8810
Epoch 2/5
491/491 [==============================] - 10s 20ms/step - loss: 0.6304 - accuracy:
0.8106 - val_loss: 0.2420 - val_accuracy: 0.9491
Epoch 3/5
491/491 [==============================] - 10s 20ms/step - loss: 0.3845 - accuracy:
0.8839 - val_loss: 0.1485 - val_accuracy: 0.9637
Epoch 4/5
491/491 [==============================] - 10s 20ms/step - loss: 0.2933 - accuracy:
0.9113 - val_loss: 0.1059 - val_accuracy: 0.9749
Epoch 5/5
491/491 [==============================] - 10s 20ms/step - loss: 0.2302 - accuracy:
0.9282 - val_loss: 0.0836 - val_accuracy: 0.9814
```

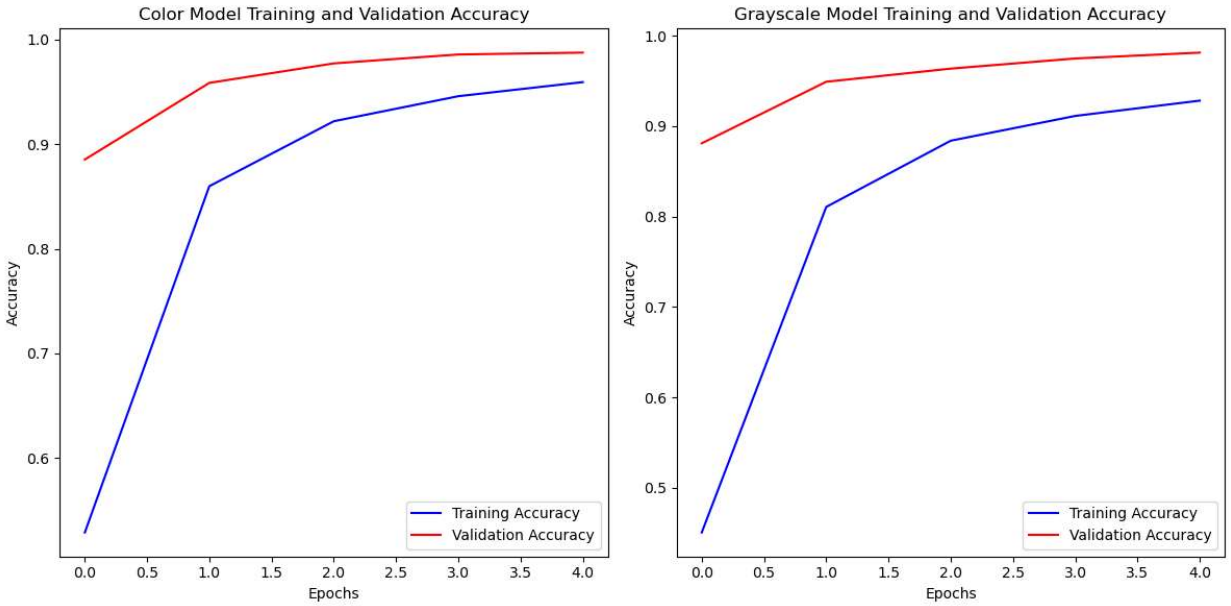
In [62]:

```
#Plot training and validation accuracy for both models
plt.figure(figsize=(12, 6))

#Plot color model accuracy
plt.subplot(1, 2, 1)
plt.plot(color_history.history['accuracy'], label='Training Accuracy', color='blue')
plt.plot(color_history.history['val_accuracy'], label='Validation Accuracy', color='re
plt.title('Color Model Training and Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

#Plot grayscale model accuracy
plt.subplot(1, 2, 2)
plt.plot(grayscale_history.history['accuracy'], label='Training Accuracy', color='blue
plt.plot(grayscale_history.history['val_accuracy'], label='Validation Accuracy', color
plt.title('Grayscale Model Training and Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.tight_layout()
plt.show()
```



In [65]:
```
#Observe precision, recall, and F1-score for all classes for both models
#Get predicted class labels
color_pred = color_model.predict(X_val_color)
color_pred_classes = np.argmax(color_pred, axis=1)
gray_pred = grayscale_model.predict(X_val_gray)
gray_pred_classes = np.argmax(gray_pred, axis=1)
true_classes = np.argmax(y_val_color, axis=1)

#Compute precision, recall, and F1-score
from sklearn.metrics import classification_report
color_report = classification_report(true_classes, color_pred_classes, target_names=cl
print("Classification report for color model:")
print(color_report)

grayscale_report = classification_report(true_classes, gray_pred_classes, target_names
print("Classification report for grayscale model:")
print(grayscale_report)
```

```
246/246 [==============================] - 1s 4ms/step
246/246 [==============================] - 1s 2ms/step
Classification report for color model:
                                precision    recall  f1-score   support

            Speed limit (20km/h)       1.00      1.00      1.00        44
            Speed limit (30km/h)       0.99      0.98      0.99       474
            Speed limit (50km/h)       0.98      0.97      0.98       452
            Speed limit (60km/h)       1.00      0.95      0.97       295
            Speed limit (70km/h)       0.98      0.99      0.99       420
            Speed limit (80km/h)       0.94      0.96      0.95       332
     End of speed limit (80km/h)       1.00      1.00      1.00        82
           Speed limit (100km/h)       0.99      0.96      0.97       276
           Speed limit (120km/h)       0.94      1.00      0.97       283
                      No passing       1.00      0.99      0.99       304
 No passing vehicle over 3.5 tons       0.99      0.99      0.99       349
 Right-of-way at the intersection       1.00      0.98      0.99       242
                  Priority road       1.00      1.00      1.00       454
                          Yield       1.00      1.00      1.00       441
                           Stop       1.00      1.00      1.00       167
                    No vehicles       1.00      1.00      1.00       127
   Vehicle > 3.5 tons prohibited       1.00      1.00      1.00        65
                       No entry       1.00      1.00      1.00       231
                General caution       1.00      0.99      1.00       238
            Dangerous curve left       1.00      0.98      0.99        49
           Dangerous curve right       0.97      0.99      0.98        78
                    Double curve       0.98      0.98      0.98        63
                      Bumpy road       0.99      1.00      0.99        86
                   Slippery road       0.95      0.99      0.97       109
       Road narrows on the right       0.97      1.00      0.98        59
                       Road work       0.98      0.99      0.99       298
                 Traffic signals       0.97      1.00      0.98       122
                     Pedestrians       1.00      1.00      1.00        47
                Children crossing       1.00      0.97      0.98        99
                Bicycles crossing       1.00      0.97      0.98        59
               Beware of ice/snow       0.97      0.96      0.96        95
            Wild animals crossing       0.99      0.99      0.99       160
      End speed + passing limits       1.00      1.00      1.00        41
                Turn right ahead       0.99      1.00      1.00       138
                 Turn left ahead       1.00      1.00      1.00        88
                      Ahead only       0.99      1.00      1.00       224
            Go straight or right       1.00      0.99      0.99        80
             Go straight or left       1.00      1.00      1.00        47
                      Keep right       1.00      1.00      1.00       418
                       Keep left       1.00      1.00      1.00        58
            Roundabout mandatory       0.98      1.00      0.99        60
               End of no passing       1.00      1.00      1.00        47
End no passing vehicle > 3.5 tons       1.00      1.00      1.00        41

                        accuracy                           0.99      7842
                       macro avg       0.99      0.99      0.99      7842
                    weighted avg       0.99      0.99      0.99      7842

Classification report for grayscale model:
                                precision    recall  f1-score   support

            Speed limit (20km/h)       1.00      0.93      0.96        44
            Speed limit (30km/h)       0.98      0.97      0.98       474
            Speed limit (50km/h)       0.97      0.99      0.98       452
            Speed limit (60km/h)       1.00      0.96      0.98       295
```

Speed limit (70km/h)	0.99	0.99	0.99	420
Speed limit (80km/h)	0.90	0.95	0.92	332
End of speed limit (80km/h)	0.99	1.00	0.99	82
Speed limit (100km/h)	0.99	0.91	0.95	276
Speed limit (120km/h)	0.96	0.98	0.97	283
No passing	1.00	0.98	0.99	304
No passing vehicle over 3.5 tons	0.99	1.00	1.00	349
Right-of-way at the intersection	0.97	0.98	0.98	242
Priority road	1.00	0.99	0.99	454
Yield	1.00	1.00	1.00	441
Stop	1.00	0.99	1.00	167
No vehicles	0.99	1.00	1.00	127
Vehicle > 3.5 tons prohibited	0.98	1.00	0.99	65
No entry	1.00	1.00	1.00	231
General caution	0.98	0.99	0.98	238
Dangerous curve left	0.96	0.98	0.97	49
Dangerous curve right	0.95	0.92	0.94	78
Double curve	0.95	1.00	0.98	63
Bumpy road	0.99	1.00	0.99	86
Slippery road	1.00	0.96	0.98	109
Road narrows on the right	0.97	0.98	0.97	59
Road work	0.95	0.99	0.97	298
Traffic signals	0.99	0.97	0.98	122
Pedestrians	1.00	0.98	0.99	47
Children crossing	0.99	0.95	0.97	99
Bicycles crossing	1.00	0.95	0.97	59
Beware of ice/snow	0.99	0.89	0.94	95
Wild animals crossing	0.97	1.00	0.98	160
End speed + passing limits	0.95	1.00	0.98	41
Turn right ahead	0.98	0.99	0.99	138
Turn left ahead	1.00	0.99	0.99	88
Ahead only	1.00	1.00	1.00	224
Go straight or right	0.97	0.97	0.97	80
Go straight or left	1.00	1.00	1.00	47
Keep right	1.00	1.00	1.00	418
Keep left	1.00	1.00	1.00	58
Roundabout mandatory	1.00	1.00	1.00	60
End of no passing	0.96	0.96	0.96	47
End no passing vehicle > 3.5 tons	1.00	1.00	1.00	41
accuracy			0.98	7842
macro avg	0.98	0.98	0.98	7842
weighted avg	0.98	0.98	0.98	7842

Both models had great classification across all classes. Nearly every precision value, recall score, and F1-score was 1.00 or just below 1.00. The lowest score was a 0.89 (a recall score for the 'Beware of ice/snow' class), but the majority of classes had scores exceeding 0.95 in all metrics. A significant amount of classes had perfect 1.00 scores in all metrics. The classes are excellent, and considering the purpose of the program, no improvement is needed.

In [67]:
```
#Evaluate model on test set
color_test_loss, color_test_accuracy = color_model.evaluate(test_color, test_labels_er
gray_test_loss, gray_test_accuracy = grayscale_model.evaluate(test_gray, test_labels_e
```

```
395/395 [==============================] - 2s 4ms/step - loss: 0.2541 - accuracy: 0.9
282
395/395 [==============================] - 1s 3ms/step - loss: 0.2813 - accuracy: 0.9
317
```

In [69]:

```
#Test set classification report
color_predictions = color_model.predict(test_color)
color_predicted_classes = np.argmax(color_predictions, axis=1)
grayscale_predictions = grayscale_model.predict(test_gray)
grayscale_predicted_classes = np.argmax(grayscale_predictions, axis=1)
test_classes = np.argmax(test_labels_encoded, axis=1)

print("Color model classification report:")
print(classification_report(test_classes, color_predicted_classes, target_names=class_

print("Grayscale model classification report:")
print(classification_report(test_classes, grayscale_predicted_classes, target_names=cl
```

```
395/395 [==============================] - 1s 4ms/step
395/395 [==============================] - 1s 3ms/step
Color model classification report:
                                precision    recall  f1-score   support

            Speed limit (20km/h)       0.97      0.95      0.96        60
            Speed limit (30km/h)       0.95      0.96      0.95       720
            Speed limit (50km/h)       0.94      0.98      0.96       750
            Speed limit (60km/h)       0.93      0.79      0.86       450
            Speed limit (70km/h)       0.96      0.95      0.96       660
            Speed limit (80km/h)       0.79      0.93      0.85       630
     End of speed limit (80km/h)       0.99      0.74      0.85       150
           Speed limit (100km/h)       0.99      0.80      0.89       450
           Speed limit (120km/h)       0.81      0.97      0.88       450
                      No passing       0.97      0.94      0.96       480
 No passing vehicle over 3.5 tons       0.97      0.99      0.98       660
 Right-of-way at the intersection       0.93      0.92      0.92       420
                  Priority road       0.99      0.94      0.97       690
                          Yield       0.99      1.00      0.99       720
                           Stop       1.00      1.00      1.00       270
                    No vehicles       0.92      0.99      0.95       210
   Vehicle > 3.5 tons prohibited       1.00      0.99      1.00       150
                       No entry       1.00      0.97      0.99       360
                General caution       0.99      0.82      0.90       390
            Dangerous curve left       0.68      1.00      0.81        60
           Dangerous curve right       0.72      0.99      0.83        90
                    Double curve       0.84      0.52      0.64        90
                      Bumpy road       0.92      0.99      0.96       120
                   Slippery road       0.63      0.82      0.71       150
       Road narrows on the right       0.95      0.78      0.85        90
                       Road work       0.96      0.90      0.93       480
                 Traffic signals       0.75      0.90      0.82       180
                     Pedestrians       0.84      0.45      0.59        60
                Children crossing       0.98      0.87      0.92       150
                Bicycles crossing       0.56      0.98      0.71        90
               Beware of ice/snow       0.88      0.57      0.69       150
            Wild animals crossing       0.89      0.95      0.92       270
      End speed + passing limits       1.00      1.00      1.00        60
                Turn right ahead       1.00      0.99      0.99       210
                 Turn left ahead       0.97      0.97      0.97       120
                      Ahead only       0.98      0.99      0.98       390
            Go straight or right       0.97      0.94      0.95       120
             Go straight or left       0.98      0.98      0.98        60
                      Keep right       0.99      0.98      0.98       690
                       Keep left       0.99      0.96      0.97        90
            Roundabout mandatory       0.77      0.92      0.84        90
               End of no passing       1.00      0.78      0.88        60
End no passing vehicle > 3.5 tons       0.92      0.99      0.95        90

                        accuracy                           0.93     12630
                       macro avg       0.91      0.90      0.90     12630
                    weighted avg       0.94      0.93      0.93     12630

Grayscale model classification report:
                                precision    recall  f1-score   support

            Speed limit (20km/h)       1.00      0.43      0.60        60
            Speed limit (30km/h)       0.89      0.96      0.92       720
            Speed limit (50km/h)       0.92      0.97      0.95       750
            Speed limit (60km/h)       0.89      0.91      0.90       450
```

Speed limit (70km/h)	0.98	0.94	0.96	660
Speed limit (80km/h)	0.81	0.93	0.86	630
End of speed limit (80km/h)	1.00	0.83	0.91	150
Speed limit (100km/h)	0.98	0.81	0.88	450
Speed limit (120km/h)	0.89	0.91	0.90	450
No passing	0.95	0.99	0.97	480
No passing vehicle over 3.5 tons	0.97	0.99	0.98	660
Right-of-way at the intersection	0.88	0.98	0.93	420
Priority road	0.98	0.98	0.98	690
Yield	0.99	1.00	0.99	720
Stop	0.97	0.96	0.96	270
No vehicles	0.96	1.00	0.98	210
Vehicle > 3.5 tons prohibited	0.99	0.99	0.99	150
No entry	0.99	0.95	0.97	360
General caution	0.94	0.84	0.89	390
Dangerous curve left	0.82	1.00	0.90	60
Dangerous curve right	0.86	0.96	0.91	90
Double curve	0.75	0.69	0.72	90
Bumpy road	0.94	0.97	0.95	120
Slippery road	0.89	0.90	0.90	150
Road narrows on the right	0.99	0.81	0.89	90
Road work	0.94	0.94	0.94	480
Traffic signals	0.94	0.81	0.87	180
Pedestrians	0.57	0.48	0.52	60
Children crossing	0.93	0.91	0.92	150
Bicycles crossing	0.81	0.97	0.88	90
Beware of ice/snow	0.90	0.47	0.62	150
Wild animals crossing	0.83	1.00	0.91	270
End speed + passing limits	0.98	1.00	0.99	60
Turn right ahead	0.93	0.99	0.96	210
Turn left ahead	0.87	0.99	0.93	120
Ahead only	0.98	0.98	0.98	390
Go straight or right	0.98	0.98	0.98	120
Go straight or left	1.00	1.00	1.00	60
Keep right	0.97	0.92	0.95	690
Keep left	0.99	0.94	0.97	90
Roundabout mandatory	0.94	0.91	0.93	90
End of no passing	0.97	0.60	0.74	60
End no passing vehicle > 3.5 tons	1.00	1.00	1.00	90
accuracy			0.93	12630
macro avg	0.92	0.90	0.90	12630
weighted avg	0.93	0.93	0.93	12630

Both models performed well on the test set. The performance was noticeably worse on the test set than the validation set, and some classes had metrics significantly below 0.80. The classes were good overall but some performed relatively poorly. Each model had a few classes with disappointing F1-scores. The color model struggled with 'Double Curve' and the grayscale model struggled with 'Speed limit (20km/h)'. Both models showed poor performance when classifying 'Pedestrians' and 'Beware of ice/snow'. That said, both models had a fairly low loss value (0.2541 for the color model and 0.2813 for the grayscale model) and both models had high accuracy scores (0.9282 for the color model and 0.9317 for the grayscale model).

Compare color and grayscale models The color model showed better performance on the validation set, and had lower training and validation loss and higher training and validation

accuracy than the grayscale model. Both models had stellar classification reports, but the color model showed slightly better scores in the three metrics. Neither model showed signs of overfitting. The models performed roughly equivalently on the test set. The grayscale model had higher accuracy, but the color model had lower loss. The performance reports were similar enough that neither model appeared to be blatantly superior, though the color model had higher metrics overall with the two classes that both models struggled with ('Pedestrians' and 'Beware of ice/snow'). Both models performed admirably, and each had weaknesses in their classifications. However, the color model seemed slighly better overall, which makes sense considering that many traffic signs are color-coded.