A SIMPLIFIED TRAFFIC SIGN DETECTION AND RECOGNITION SYSTEM USING CONVOLUTIONAL NEURAL NETWORKS

Conference Paper · July 2022

CITATIONS
0

READS
80

2 authors:

Abubakar Sadiq Muhammad
School of technology Kano
27 PUBLICATIONS **76** CITATIONS
SEE PROFILE

Fahad Abdu Jibrin
Ahmadu Bello University
10 PUBLICATIONS **110** CITATIONS
SEE PROFILE

3rd International Conference on Applied Engineering and Natural Sciences

https://www.icaens.com/

July 20-23, 2022, Konya, Turkey

A SIMPLIFIED TRAFFIC SIGN DETECTION AND RECOGNITION SYSTEM USING CONVOLUTIONAL NEURAL NETWORKS

Abubakar Sadiq Muhammad*, Fahad Abdu Jibrin [2]

[1] *Department of Computer Engineering, Kano State Polytechnic, Kano, Nigeria*
[2]*Department of Telecommunication Engineering Ahamdu Bello University, Zaria,, Nigeria*

(alsiddiq4uhd@gmail.com) Email of the corresponding author

Abstract – The development in automotive intelligent technology in ADAS (Advanced Driver Assistance System) has made Traffic sign detection and recognition play an important role in expert systems. It instantly assists drivers or automatic driving systems in detecting and recognizing traffic signs effectively. This work proposed a method to design real-time traffic sign detection and recognition system in a real traffic situation. The images of the road scene were converted to grayscale images and then filtered the grayscale images with simplified color segmentation techniques, where the parameters were optimized. Convolution Neural networks and support vector machines (SVM) were employed for the detection and classification of the traffic signs. An accuracy of about 95.3% was achieved as a comparable performance with the state-of-the-art method.

Keywords – Traffic signs recognition, Colour space, Convolutional Neural Network (CNN) and Support Vector Machines

I. INTRODUCTION

Human factor remains the most common cause of road mortality. Indeed, the potentially dangerous choices made by the driver might be intentional (speed driving, for example) as they might be the result of physical tiredness, drowsiness or a poor perception and interpretation of seen scenes [1][2]. The introduction of autonomous vehicles is certainly a way to reduce these causes or even make them disappear. As part of the development of these autonomous vehicles, particularly driving assistance systems, several manufacturers and laboratories have geared their works towards the exploitation of visual information because of its usefulness for the detection of road, vehicles, pedestrians and traffic signs. Advance Driving Assistance System (ADAS) is a principle of road signs recognition to detect signs, interpret their meaning, then transmit the information to the driver (by a projection on a windshield, a screen or a smartphone) or even better, transmit the information to the vehicle that carries out the execution without needing a human decision[3]. This development in automotive intelligent technology had made many car companies to invest in ADAS (Advanced Driver Assistance System) research. A keen area of concern is the TSR (Traffic Sign Recognition) systems meant to remind drivers to pay attention to the speed. With the increasing demand for the intelligence of vehicles, it is becomes paramount to detect and recognize traffic signs automatically through computer technology. Road and traffic signs are those that use a visual/symbolic language about the road(s) ahead that can be interpreted by drivers. These terms are used interchangeably are keen to significantly improve the safety and implementation of autonomous driving. Traffic signs are often designed to be of a particular shape and colour with symbols inside so that there is a significant difference between the traffic signs and the background[4]. Among the major setbacks for traffic regongition include (i) The colors of road signs, particularly red, may fade after long exposure tothe sun[5].

Moreover, paint may even flake or peel off. (ii) Air pollution and weather conditions (e.g., rain, fog, shadows, and clouds) may decrease the visibility of road signs.[6] (iii) Outdoor lighting conditions varying from day to night may affect the colors of road signs. (iv) Obstacles, such as vehicles, pedestrians, and other road signs, may partially occlude road signs [7]. (v) As vehicle mounted cameras are not always perpendicular to the traffic signs, and the shape of traffic signs are often distorted in road scenes, the shape information of traffic signs is no longer fully reliable as Video images of road signs will have motion blur if the camcorder is mounted on a moving vehicle due to vehicle vibration as well as motion[8]. Previous researches usually consider undistorted and completely visible models of the traffic signs for recongition which is not the case for real characteristics of the road environment[9]. For this reason, current researches move towards the development of recognition systems with close proximity to real images of road signs that do not appear like their models. Recently deep learning has been employed for action recognition since the breakthrough [10], several works have been trying to design effective deep network architecture for task of recognition in images and videos. Among the existing approaches for palmprint verification, convolutional neural networks (CNNs) outperform the state-of-the-art techniques. CNNs are a class of deep learning concerned with architectures inspired by the structure and function of the brain, called artificial neural networks. Wellknown CNN architectures such as Alexnet [10], VGGNet [11], and ResNet [12] have shown excellent performance in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) competitions [13]. The CNN possess three important properties: Sparse interaction, Parameter sharing and Equivalent representation [14]. It has shown competency in extracting features from signals making its provide promising results in image classification. After convolution, there are usually pooling and fully connected layers that perform the classification; these convolution pooling is common in CNN where max or average pooling is carried out [15] making it possess the advantage of speeding up the training process in large dataset [16].

This work presents a system to detection and classification of road and traffic signs. The system is based on three major stages: traffic sign detection(based on the shape and color attributes of traffic signs) and traffic sign recognition which often employs convolutional neural networks (CNNs) and SVM known for ability to come up with more discriminative features and clear distintion between classes respectively. The overall framework is divided into six sections with colour space employed for image segmentation explained in section three; Section four presents the proposed approach and the various modules to generate the Traffic recogniton system. The result, discussion and conclusions are given in section five and six.

II. LITERATURE REVIEW

Human beings are capable of distinguishing traffic signs from a background as they possess certain unique scolor scheme, which includes red, blue, and white that allow for distinguishing traffic signs from the background scene thus, for a computer detection system, color information is also an important feature[4]. Traffic detection can be done using both gray-scale and color based, Gray-scale based has the search mainly based on shape and can be quite expensive in terms of computational time [17], [18] . with the color based detection , the search is based on color via color segmentation which appears to be faster than a shape detection, eventhough requiring additional filtering but the developed system is more robust, faster, straightforward and most simplest method [19–22]. Most of these methods have been developed using color base trasformations as red, green, blue image (RGB) is too sensitive to illumination, color space conversion. The HSV/HSI color space is the most used [8][23] but other color spaces, such as CIECAM97 [24], can be used as well. These spaces are used because chromatic information can be easily separated from the lighting information: this is used to detect a specified color in almost all light conditions. Shadeed et al. [25] segmented the image by the U and V chromaticity channels in YUV space, where U is positive and V is the color red. This information was combined with the hue channel of the hue, saturation, value (HSV) color space to segment the red road signs from the image

In [26], thresholds were used in the color channel of the HSV color space, to segment the red road signs. Unfortunately, due to strong light, poor light, and other bad weather conditions, color-based detection methods often fail to achieve better results.

Owing to the fact that traffic signs from most countries have a strict shape and symbol distinction.with pairs of different signs when converted to grayscale which appear the same, This design allows color blind and color weak drivers and pedestrians to recognize traffic signs without or with less color information. Based on such features, some algorithms completely ignore color information and adopt shape-based image segmentation [27–29]. In addition to the above-mentioned factors, some algorithms use image features other than the raw color space and shape information. The authors in Reference [30] used the color probability model, and traffic sign proposals were then extracted by finding the most stable extremal regions on these maps. [31] used rotation invariant binary pattern-based features. In any case, finding good feature expression is an important part of traffic sign detection, and is also an important research topic in this field.

Recently, as deep learning methods have demonstrated prominent representation capacity and achieved outstanding performance in traffic sign recognition, more scholars have applied technologies to this area [32–35]. The author of [32] used a convolution neural network based on deep learning with a hinge loss stochastic gradient descent method, which achieved a high recognition rate. Dan [33] provided a multi-column deep neural network for traffic classification running on a graphic processing unit (GPU), and obtained a better-than-human recognition rate. Qian [34] used CNN as the classifier to learn the discriminative feature of max pooling positions to classify traffic signs and obtained a comparable performance with the-state-of-the-art method. Ali [35] used a procedure based on color segmentation, histogram of oriented gradients, and CNN, for the detection and recognition of traffic signs, which achieved better classification accuracy and computational speed. For this work, the detection process involves the use of colour segmentation followed by shape detection and finally classification by Convolutional Neural Networks and Support vector machines to further study the choice of discriminative features and to research the network structure to improve classification accuracy and processing time.

III. COLOUR SPACE

Color is a powerful descriptor that often simplifies object identification and extraction from a scene. When humans are asked to describe a picture, they generally give a list of objects within the picture as well as their relative positions [36]. A typical color image capturing system relies on a trichromatic input based on the additive primary colors red, green and blue [37][38]. This is commonly known as the RGB color space. The RGB model corresponds most closely to the physical sensors for colored light (e.g., the cones in the human eye), and it is implemented as red, green, and blue filters in most color CCD sensors. However, the human perception of color qualities can be said to follow more closely the HSI (Hue, Saturation, and Intensity) model [39]. This model creates a color image representation based on the amount of light (i.e., the Intensity), the amount of color (i.e., the Saturation) and color as described by the wavelength (i.e., the Hue). Humans perceive color as a result of light in the visible region of the spectrum (i.e., having wavelengths in the region of 400 nm to 700 nm [40]) being projected upon the retina. Therefore, color is the brain's reaction to a specific visual stimulus (i.e., an object reflecting light of a certain wavelength). Although color can be precisely described by measuring its spectral power distribution or SPD (the intensity of the visible electromagnetic radiation at many discrete wavelengths) this leads to a large degree of redundancy. The human retina has only three types of color photoreceptor cone cells each of which responds to incident radiation with a somewhat different spectral response curve. These three broad spectral bands roughly correspond to what humans perceive as red, green and blue light. The rod is the fourth type of

photoreceptor cell present in the retina. Rods are effective only at extremely low light levels. The signals from these color sensitive cells (cones), together with those from the rods (sensitive to intensity only), combine in the brain to give "sensations" of different colors [41].

(i) RGB

The RGB (Red, Green, Blue) space is used most frequently in computer graphics and image processing applications. A color in this space is represented by a triplet of values typically between zero and one and is usually scaled by 255 for an 8-bit representation. Each color can be broken down into its relative intensity in the three primaries corresponding to the spectral response of one of the three types of cones present in the human eye: red, green and blue. The space is easily represented as a three dimensional cube where each axis represents the strength of the color in one of the three primaries.

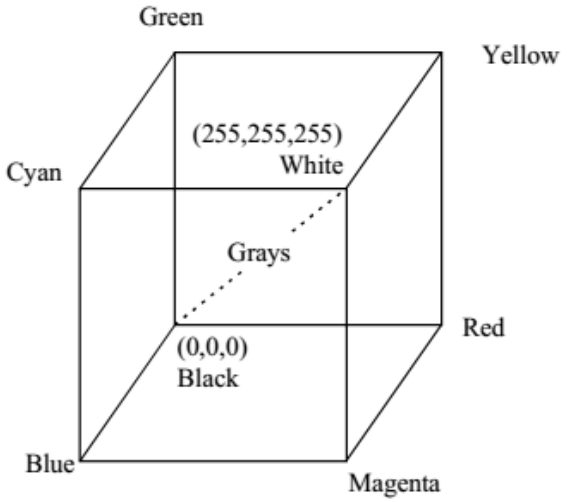

Figure 1: RGB Colour Space

(ii) HIS

The HSI (Hue, Saturation, and Intensity) model [37] can be said to follow more closely the human perception of color qualities. Hue (H) is the color as described by wavelength – for example, the distinction between red and blue. Hue represents the fundamental or dominant color. Saturation (S) represents the amount of a color present, where pastel shades (e.g. pink) have low saturation values while pure spectral colors (e.g. red) are completely saturated. The intensity (I) represents the overall brightness or the amount of light. It is independent of color and is a linear value. It is measured as an angle on a color circle with the three primary colors spaced 120◦apart. The first two values specify the chromaticity of a color point. Figure 2 shows the relationship between the HSI and RGB spaces.

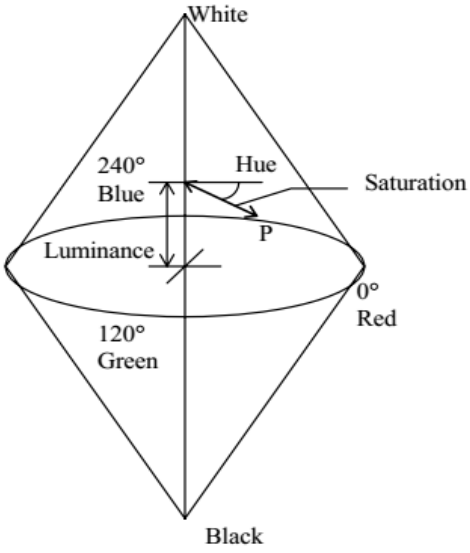

Figure 2: HSI Colour Space

It is noted that the HSI color space is one of several spaces that can be derived from the General Hue, Luminance and Saturation (GHLS) space. There are other slightly different interpretations of hue, luminance and saturation. There is a general transformation from RGB to GHLS [39]. By setting certain parameters in this transformation, one can specify any of the transformations from the RGB space to an HSI, HLS or HSV [39] (Hue, Saturation and Value) representation. The HSI definition is given in [36-40] as shown below:

$$\cos\alpha = \frac{(R-G)+(R-B)}{2[(R-G)^2+(R-B)(G-B)^1\,]^{1/2}}$$

$$H = \begin{cases} \alpha & if\ B < G \\ 2\pi - \alpha & Otherwise \end{cases}$$

$$S = 1 - \frac{3.min(R,G,B)}{R+G+B}$$

$$I = \frac{1}{3}(R+G+B)$$

The plane I=1 (for normalized R, G, B values) defines the hue triangle (with vertices at the extremities of the Red, Green and Blue axes) from which these equations have been derived. Shafer's model was applied to the different components of the HSI space [37].

IV. PROPOSED APPROACH FOR THE TRAFFIC SIGN DETECTION AND RECOGNITION

A. RECOGNITION MODULE

A system to detect and recognize road and traffic signs should be able to work in two modes; the training mode in which a database can be built by collecting a set of traffic signs for training and validation, and a prediction mode in which the system can recognize a traffic sign which has not been seen before. It consists of a number of modules which work together to perform this recognition as shown in figure 3. These modules are as follows: *The image, Colour Segmentation , Shape Analysis, Raw Image Database, Training Database, Feature Extraction, Classification.*

a. **The image:** A digital image which is clear and sharp with different sizes is necessary. Images presented at this stage are used in later stages to develop and validate the colour segmentation algorithm, the recognition stage, and to build the classification system.

b. **Colour Segmentation:** Colour segmentation is an important step employed to eliminate all background objects and insignificant information in the image. It generates a binary image containing the road signs and any other objects similar to the colour of the road sign. This step reduces the amount of calculation needed in the following steps as it radically reduces the number of probable objects. A colour segmentation algorithm should be robust enough to work in a wide spectrum of environmental conditions and be able to generate binary images even when traffic sign colours are attenuated.

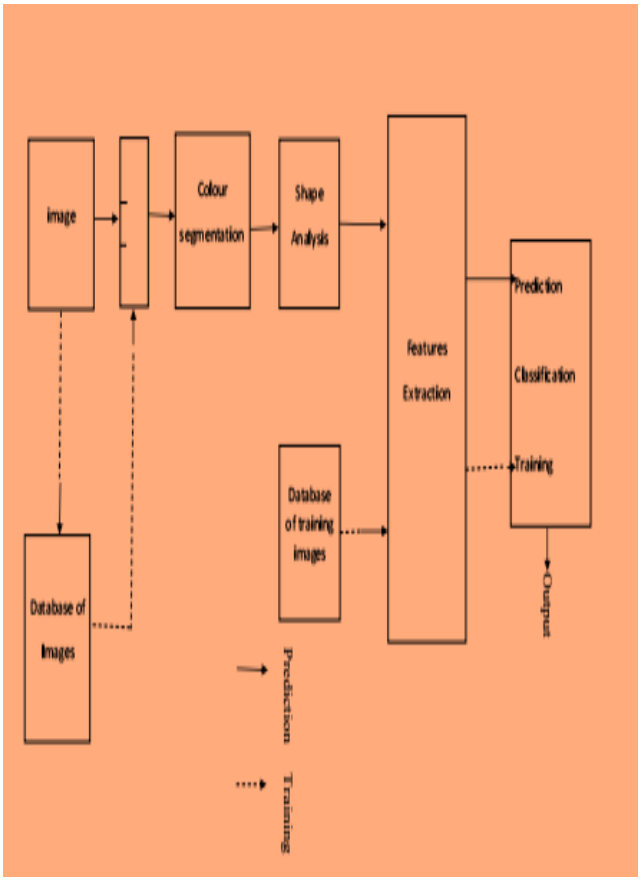

Figure 3: Block diagram of Traffic Sign Detection and Recogntion System

c. **Shape Analysis:** The task of this module includes cleaning the binary image from noise and small objects, applying connected components labelling algorithm, and recognizing the traffic sign. This module normalizes the recognized traffic sign so that it becomes invariant to the in-plane transformations meaning that the resultant sign has a fixed size and it is located in a standard position where its center of gravity is located in the center of the image. This module works in two modes; the training and the prediction mode. In the training mode it is invoked to create or update the training image database. In the prediction mode it prepares every object in the binary image to be in standard format and ready for feature extraction.

d. **Raw Image Database:** This database is simply a collection of traffic scenes gathered by the camera. Images in this database are categorized according to the type of the sign, condition of the sign such as occluded, damaged, or faded, weather conditions, light geometry, and defects in the images such as blurring. The database represents the main

source of images from which another database called the "training database" is built. The number of images in this database should be extensive enough to cover the large variety of traffic signs which are used for training the system. In addition to this, the number of images containing the same type of sign should be vast to provide a good number of examples for training. An Access database program manages the images and the information about them so that selecting images with certain characteristics is straightforward.

e. Training Database: The training database consists of binary images of a normalized size such as 30x30 pixels. The database is created and updated in the training mode in such a way that binary images of the desired traffic signs are selected from a set of images. This database is used either directly or by extracting some features to train and validate the classifier. In the prediction mode, the database is invoked to train the classifier before any classification task takes place.

f. Feature Extraction: This module contains algorithms which are used to extract features from either the training images in the training database or images directly from the shape analysis unit. It allows the classifier to be trained by either binary images or by features. The CNN was employed for this task..

g. Classification: this stage proivde a clear distintion between the signs . The algorithm learns a separating hyperplane to maximize the margin and to produce good generalization ability. İn this work, SVM was employed for the classifcation due to the good generalization performance on a large number of real-life data and due to the fact that the approach is properly motivated, it has been used for a wide range of applications.

V. RESULTS AND DISCUSSIONS

Our approach to building this traffic sign classification model is discussed in four steps:

i. Explore the dataset
ii. Build a CNN model
iii. Train and validate the mode
iv. Test the model with test dataset.

To ensure smooth processing the **TK GUI** toolkit was employed. Tkinter is a python platform binding to the. TK GUI toolkit It is the standard Python interface to the TK GUI toolkit, and is python's de facto standard GUI. Tkinter is included with standard Linux,Microsoft windows and Mac OS X installs of Python. For this work that of Microsoft windows was used.

Step 1: Explore the dataset

The database employed for this work is the German Traffic Sign Recognition Benchmark (GTSRB) databases .Our 'train' folder contains 43 folders each representing a different class. The range of the folder is from 0 to 42. With the help of the OS module, we iterate over all the classes and append images and their respective labels in the data and labels list.

```
[9]: import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import tensorflow as tf
     from PIL import Image
     import os
     from sklearn.model_selection import train_test_split
     from keras.utils import to_categorical
     from keras.models import Sequential
     from keras.layers import Conv2D, MaxPool2D, Dense, Flatten, Dropout

     data = []
     labels = []
     classes = 43
     cur_path = os.getcwd()

     for i in range(classes):
         path = os.path.join(cur_path,'train',str(i))
         images = os.listdir(path)

         for a in images:
             try:
                 image = Image.open(path + '\\'+ a)
                 image = image.resize((30,30))
                 image = np.array(image)
                 #sim = Image.fromarray(image)
                 data.append(image)
                 labels.append(i)
             except:
                 print("Error loading image")
     data = np.array(data)
     labels = np.array(labels)
```

Figure 4: Importing Packages and Datasets

The PIL library was used to open image content into an array. We have stored all the images and their labels into lists (data and labels). A conversion of the list into numpy arrays was done for feeding to the model.The shape of data is (39209, 30, 30, 3) which means that there are 39,209 images of size 30×30 pixels and the last 3 means the data contains colored images (RGB value).

```
[10]: print(data.shape, labels.shape)
      X_train, X_test, y_train, y_test = train_test_split(data, labels, test_size=0.2, random_state=42)


      print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)


      y_train = to_categorical(y_train, 43)
      y_test = to_categorical(y_test, 43)

(39209, 30, 30, 3) (39209,)
(31367, 30, 30, 3) (7842, 30, 30, 3) (31367,) (7842,)
```

Figure 5: Splitting Test and Train Data

With the sklearn package, we use the train_test_split() method to split training and testing data. With the aid of keras.utils package, we use categorical method to convert the labels present in y_train and t_test into one-hot encoding.

Step 2: Build CNN Model

To classify the images into their respective categories, we will build a CNN model CNN is best for image classification purposes. The architecture of our model is:

- **2 Conv2D layer (filter=32, kernel_size=(5,5), activation="relu")**
- **MaxPool2D layer (pool_size=(2,2))**
- **Dropout layer (rate=0.25)**
- **2 Conv2D layer (filter=64, kernel_size=(3,3), activation="relu")**
- **MaxPool2D layer (pool_size=(2,2))**
- **Dropout layer (rate=0.25)**
- **Flatten layer to squeeze the layers into 1 dimension**
- **Dense Fully connected layer (256 nodes, activation="relu")**
- **Dropout layer (rate=0.5)**
- **Dense layer (43 nodes, activation="softmax")**

We compile the model with Adam optimizer which performs well and loss is "categorical_crossentropy" because we have multiple classes to categorize.

```
[11]: model = Sequential()
      model.add(Conv2D(filters=32, kernel_size=(5,5), activation='relu', input_shape=X_train.shape[1:]))
      model.add(Conv2D(filters=32, kernel_size=(5,5), activation='relu'))
      model.add(MaxPool2D(pool_size=(2, 2)))
      model.add(Dropout(rate=0.25))
      model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu'))
      model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu'))
      model.add(MaxPool2D(pool_size=(2, 2)))
      model.add(Dropout(rate=0.25))
      model.add(Flatten())
      model.add(Dense(256, activation='relu'))
      model.add(Dropout(rate=0.5))
      model.add(Dense(43, activation='softmax'))


      #Compilation of the model
      model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Figure 6: Building CNN Model

Step 3: Train and Validate the Model

After building the model architecture, we then train the model using model.fit(). I tried with batch size 32 and 64. Our model performed better with 64 batch size. And after 15 epochs the accuracy was stable.

```
[12]: epochs = 15
      history = model.fit(X_train, y_train, batch_size=64, epochs=epochs,validation_data=(X_test, y_test))

      Train on 31367 samples, validate on 7842 samples
      Epoch 1/15
      31367/31367 [==============================] - 82s 3ms/step - loss: 2.3108 - accuracy: 0.4369 - val_lo
      ss: 0.6590 - val_accuracy: 0.8234
      Epoch 2/15
      31367/31367 [==============================] - 82s 3ms/step - loss: 0.8266 - accuracy: 0.7606 - val_lo
      ss: 0.3468 - val_accuracy: 0.9100
      Epoch 3/15
      31367/31367 [==============================] - 83s 3ms/step - loss: 0.5738 - accuracy: 0.8283 - val_lo
      ss: 0.1882 - val_accuracy: 0.9504
      Epoch 4/15
      31367/31367 [==============================] - 85s 3ms/step - loss: 0.4282 - accuracy: 0.8720 - val_lo
      ss: 0.1373 - val_accuracy: 0.9661
      Epoch 5/15
      31367/31367 [==============================] - 84s 3ms/step - loss: 0.3565 - accuracy: 0.8950 - val_lo
      ss: 0.1068 - val_accuracy: 0.9702
      Epoch 6/15
      31367/31367 [==============================] - 81s 3ms/step - loss: 0.3081 - accuracy: 0.9074 - val_lo
      ss: 0.1527 - val_accuracy: 0.9575
      Epoch 7/15
      31367/31367 [==============================] - 81s 3ms/step - loss: 0.2730 - accuracy: 0.9192 - val_lo
      ss: 0.0888 - val_accuracy: 0.9753
      Epoch 8/15
      31367/31367 [==============================] - 81s 3ms/step - loss: 0.2429 - accuracy: 0.9271 - val_lo
      ss: 0.0934 - val_accuracy: 0.9737
      Epoch 9/15
      31367/31367 [==============================] - 84s 3ms/step - loss: 0.2429 - accuracy: 0.9299 - val_lo
      ss: 0.0772 - val_accuracy: 0.9763
      Epoch 10/15
      31367/31367 [==============================] - 81s 3ms/step - loss: 0.2176 - accuracy: 0.9364 - val_lo
      ss: 0.1133 - val_accuracy: 0.9663
      Epoch 11/15
      31367/31367 [==============================] - 82s 3ms/step - loss: 0.2200 - accuracy: 0.9360 - val_lo
      ss: 0.0823 - val_accuracy: 0.9786
      Epoch 12/15
      31367/31367 [==============================] - 80s 3ms/step - loss: 0.2046 - accuracy: 0.9406 - val_lo
      ss: 0.0806 - val_accuracy: 0.9787
      Epoch 13/15
      31367/31367 [==============================] - 80s 3ms/step - loss: 0.1876 - accuracy: 0.9452 - val_lo
      ss: 0.0569 - val_accuracy: 0.9852
      Epoch 14/15
      31367/31367 [==============================] - 81s 3ms/step - loss: 0.2007 - accuracy: 0.9430 - val_lo
      ss: 0.0629 - val_accuracy: 0.9811
      Epoch 15/15
      31367/31367 [==============================] - 81s 3ms/step - loss: 0.1914 - accuracy: 0.9463 - val_lo
      ss: 0.0676 - val_accuracy: 0.9813
```

Figure 7: Train and Validation of Model

Our model got a 95% accuracy on the training dataset. With matplotlib, we plot the graph for accuracy and the loss.

```
[13]: plt.figure(0)
      plt.plot(history.history['accuracy'], label='training accuracy')
      plt.plot(history.history['val_accuracy'], label='val accuracy')
      plt.title('Accuracy')
      plt.xlabel('epochs')
      plt.ylabel('accuracy')
      plt.legend()

      plt.figure(1)
      plt.plot(history.history['loss'], label='training loss')
      plt.plot(history.history['val_loss'], label='val loss')
      plt.title('Loss')
      plt.xlabel('epochs')
      plt.ylabel('loss')
      plt.legend()

[13]: <matplotlib.legend.Legend at 0x24eece89e48>
```

Figure 8: Accuracy Plotting using Python

```
[13]: <matplotlib.legend.Legend at 0x24eece89e48>
```

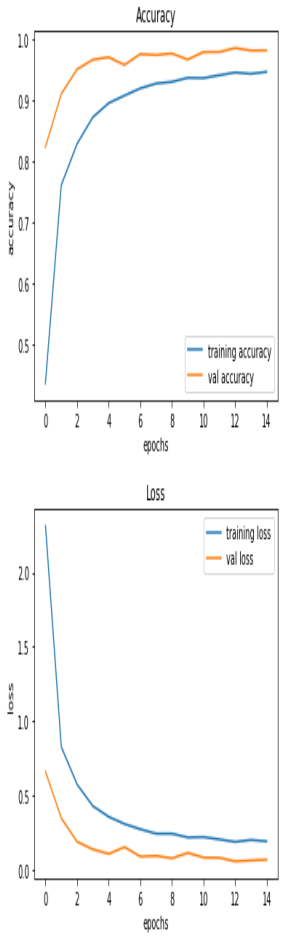

Figure 9: Accuracy and Loss Graph

Step 4: Test our Model with Test Dataset

Our dataset contains a test folder and in a test.csv file, we have the details related to the image path and their respective class labels. We extract the image path and labels using pandas. Then to predict the model, we have to resize our images to 30×30 pixels and make a numpy array containing all image data. From the sklearn.metrics, we imported the accuracy_score and observed how our model predicted the actual labels. We achieved a 95% accuracy in this model. In the end, we are going to save the model that we have trained using the Keras model.save() function as shown in figure

```
[14]: from sklearn.metrics import accuracy_score
      import pandas as pd
      y_test = pd.read_csv('Test.csv')

      labels = y_test["ClassId"].values
      imgs = y_test["Path"].values

      data=[]

      for img in imgs:
          image = Image.open(img)
          image = image.resize((30,30))
          data.append(np.array(image))

      X_test=np.array(data)

      pred = model.predict_classes(X_test)

      #Accuracy with the test data
      from sklearn.metrics import accuracy_score
      accuracy_score(labels, pred)

[14]: 0.9532066508313539
```

```
model.save('traffic_classifier.h5')
```

Figure 10: Testing Model with Test data

Step 5: Traffic Signs Classifier GUI

Now we built a graphical user interface for our traffic signs classifier with Tkinter. Tkinter is a GUI toolkit in the standard python library. we have first loaded the trained model 'traffic_classifier.h5' using Keras. And then we build the GUI for uploading the image and a button is used to classification which is called the classify() function. The classify() function involves converting the image into the dimension of shape (1, 30, 30, 3). This is because to predict the traffic sign we have to provide the same dimension we have used when building the model. Then we predict the class, the model.predict_classes(image) returns us a number between (0-42) which represents the class it belongs to. We use the dictionary to get the information about the class. This has allowed us successfully classified the traffic signs classifier with 95% accuracy and also visualized how our accuracy and loss changes with time.

Figure 11: Examples of the detection and recognition results

Results should be clear and concise. The most important features and trends in the results should be described but should not interpreted in detail.

VI. CONCLUSION

This work has demonstrated a means of designing a road traffic sign recognition system. CNN was used for feature extraction with support vector machines employed for classification and validation. An accuracy of about 95% is seen to have been achieved

REFERENCES

[1] Y. Nguwi and A. Kouzani, "A Study on Automatic Recognition of Road Signs," in Procs. IEEE Conference on Cybernetics and Intelligent Systems, Bangkok, Thailand, June 2006, pp. 1–6.

[2] C. Fang, C. Fuh, S. Chen, and P. Yen, "A road sign recognition system based on dynamic visual model," presented at The 2003 IEEE Computer Society Conf. Computer Vision and Pattern Recognition, Madison, Wisconsin, 2003.

[3] C. Fang, S. Chen, and C. Fuh, "Road-sign detection and tracking," IEEE Trans. on Vehicular Technology, vol. 52, pp. 1329-1341, 2003.

[4] Faming Shao , Xinqing Wang *, Fanjie Meng, Ting Rui, Dong Wang and Jian Tang ," Real-Time Traffic Sign Detection and Recognition Method Based on Simplified Gabor Wavelets and CNNs" *Sensors, MDPI, Vol 18 pp 3192* , 2018, *18*, 3192

[5] S. Vitabile, A. Gentile, and F. Sorbello, "A neural network based automatic road sign recognizer," presented at The 2002 Inter. Joint Conf. on Neural Networks, Honolulu, HI, USA, 2002.

[6] J. Miura, T. Kanda, and Y. Shirai, "An active vision system for real-time traffic sign recognition," presented at 2000 IEEE Intelligent Transportation Systems, Dearborn MI, USA, 2000.

[7] S. Vitabile, A. Gentile, G. Dammone, and F. Sorbello, "Multi-layer perceptron mapping on a SIMD architecture," presented at The 2002 IEEE Signal Processing Society Workshop, 2002.

[8] S. Vitabile, G. Pollaccia, G. Pilato, and F. Sorbello, "Road sign Recognition using a dynamic pixel aggregation technique in the HSV color space," presented at 11th Inter. Conf. Image Analysis and Processing, Palermo, Italy, 2001.

[9] P. Paclik and J. Novovicova, "Road sign classification without color information," presented at Sixth Annual Conf. of the Advanced School for Computing and Imaging, Lommel, Belgium, 2000

[10] A. Krizhevsky, I. Sutskever , G.E. Hinton," Imagenet classification with deep convolutional neural networks", *Adv. Neural Inf. Process. Syst*. **V**ol**.** *25*, pp. 1097–1105, 2012.

[11] K. Simonyan, A. Zisserman, "Very deep convolutional networks for large-scale image recognition", **2014**

[12] K. He, X. Zhang, S. Ren, J. Sun, " Deep residual learning for image recognition" , IEEE Conference on Computer Vision and Pattern, Recognition, June 2016, Las Vegas, NV, USA.

[13] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh,, S.Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein et al., " Imagenet large scale visual recognition challenge", *International Journal of Computer Vision, Vol.115,pp*. 211–252, 2015.

[14] Y. LeCun, Y. Bengio, G. Hinton, "Deep learning Nature", Vol. 521, pp. 436–444, 2015.

[15] S. Ha, J. Yun, S. Choi," Multi-modal convolutional neural networks for activity recognition, IEEE International Conference on Systems, Man and Cybernetics (SMC), pp. 3017–3022, 2015.

[16] Y. Bengio," Deep learning of representations", International Conference on Statistical Language and Speech Processing, Springer. pp. 1–37, 2013

[17] D. Gavrila, "Traffic Sign Recognition Revisited," in Procs. of the 21st *DAGM Symposium fr Mustererkennung*, Bonn, Germany, 1999, pp. 86–93.

[18] G. Loy and N. Barnes, "Fast Shape-based Road Signs Detection for a Driver Assistance System," in *Procs. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, Sendai, Japan, pp. 70–75, Sept. 2004.

[19] Soendoro, W.D.; Supriana, I. Traffic sign recognition with Color-based Method, shape-arc estimation and SVM. In Proceedings of the 2011 International Conference on Electrical Engineering and Informatics, Bandung, Indonesia, 17–19 July 2011.

[20] Li, H.; Sun, F.; Liu, L.; Wang, L. A novel traffic sign detection method via color segmentation and robust shape matching. Neurocomputing **2015**, 169, 77–88. [CrossRef]

[21] Bahlmann, C.; Zhu, Y.; Ramesh, V.; Pellkofer, M.; Koehler, T. A system for traffic sign detection, tracking, and recognition using color, shape, and motion information. In Proceedings of the 2005 IEEE Intelligent Vehicles Symposium, Las Vegas, NV, USA, 6–8 June 2005.

[22] Ardianto, S.; Chen, C.; Hang, H. Real-time traffic sign recognition using color segmentation and SVM.In Proceedings of the 2017 International Conference on Systems, Signals and Image Processing (IWSSIP), Poznan, Poland, 22–24 May 2017.

[23] A. de la Escalera, J. M. Armignol, and M. Mata, "Traffic sign recognition and analysis for intelligent vehicles," Image and Vision Computing, vol. 21, no. 3, pp. 247–258, Mar. 2003.

[24] X. Gao, N. Shevtsova, K. Hong, S. Batty, L. Podladchikova, A. Golovan, D. Shaposhnikov, and V. Gusakova, "Vision Models Based Identification of Traffic Signs," in Procs. of Europ. Conf. on Color in Graphics Image and Vision, Poitiers,France, pp. 47–51, Apr. 2002,

[25] Shadeed, W.G.; Abu-Al-Nadi, D.I.; Mismar, M.J. Road traffic sign detection in color images. In Proceedings of the 10th IEEE International Conference on Electronics, Circuits and Systems, Sharjah, UAE, 14–17 December 2003.

[26] Malik, R.; Khurshid, J.; Ahmad, S.N. Road sign detection and recognition using colour segmentation, shape analysis and template matching. In Proceedings of the 2007 International Conference on Machine Learning and Cybernetics, Hong Kong, China, 19–22 August 2007.

[27] Paclk, P.; Novovicova, J. Road sign classification without color information. In Proceedings of the 6th Conference of Advanced School of Imaging and Computing, July 2000;. Available online: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.11.3982 (accessed on 7 July 2000).

[28] Jin, J.; Fu, K.; Zhang, C. Traffic sign recognition with hinge loss trained convolutional neural networks. IEEE Trans. Intell. Transp. Syst. **2014**, 15, 1991–2000. [CrossRef]

[29] Cireşan, D.; Meier, U.; Masci, J.; Schmidhuber, J. Multi-column deep neural network for traffic sign classification. Neural Netw. **2012**, 32, 333–338. [CrossRef] [PubMed]

[30] Qian, R.; Yue, Y.; Coenen, F.; Zhang, B. Traffic sign recognition with convolutional neural network based on max pooling positions. In Proceedings of the 12th International Conference on Natural Computation; Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), Changsha, China, 13–15 August 2016

[31] Youssef, A.; Albani, D.; Nardi, D.; Bloisi, D.DFast traffic sign recognition using color segmentation and deep convolutional networks. In Proceedings of the ACIVS 2016: Advanced Concepts for Intelligent Vision Systems, Lecce, Italy, 24–27 October 2016.

[32] Riveiro, B.; Díaz-Vilariño, L.; Conde-Carnero, B.; Soilán, M.; Arias, P. Automatic segmentation and shape-based classification of retro-reflective traffic signs from mobile LiDAR data. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **2016**, 9, 295–303. [CrossRef]

[33] Yang, Y.; Luo, H.; Xu, H.; Wu, F. Towards real-time traffic sign detection and classification. IEEE Trans. Actions Intell. Transp. Syst. **2016**, 17, 2022–2031. [CrossRef]

[34] Yin, S.; Ouyang, P.; Liu, L.; Guo, Y.; Wei, S. Fast traffic sign recognition with a rotation invariant binary pattern based feature. Sensors 15, 2161–2180, **2015**,. [CrossRef] [PubMed]

[35] Sheikh, D.M.A.A.; Kole, A.; Maity, T. Traffic sign detection and classification using colour feature and neural network. In Proceedings of the 2016 International Conference on Intelligent Control

Power and Instrumentation (ICICPI), Kolkata, India, 21–23 October 2016

[36] A.S.Muhammad, A.L.Musa , A.U. Umar,"Image edge detection using YUV color space: An alternative to RGB Color space", International Journal of Advanced Academic Research, IJAAR, Vol. 6, No7, July, 2020.

[37] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Addison-Wesley, 1993

[38] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision*, vol. 1, AddisonWelsey: Reading, MA, 1992

[39] Anil K. Jain. *Fundamentals of Digital Image Processing*, Prentice Hall, Inc., Englewood Cliffs, NJ, 1989.

[40] Gudrun J. Klinker, Steven A. Shafer and Takeo Kanade, "A Physical Approach to Color Image Understanding," *International Journal of Computer Vision*, Vol. 4, No. 1, pp. 7-38, 1990.

[41] Gudrun J. Klinker, Steven A. Shafer and Takeo Kanade, "The measurement of highlights in color images," *International Journal of Computer Vision*, Vol. 2, No. 1, pp. 7-32, 1988.

[42] F. Kurugöllü, and B. Sankur, "Color Image Segmentation Based on Multithresholding and Fusion," in *Proc. ICIP*, October 1997.

[43] Levkowitz, Haim, *Color Theory and Modeling for Computer Graphics, Visualization, and Multimedia Applications*. Kluwer Academic Publishers: Norwell MA, USA, 1997.

[44] J. Liu and Y.-H. Yang, "Multiresolution Color Image Segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 16, no. 7, pp. 689-700, July 1994.

[45] Q.-T. Luong, "Color in computer vision," *Handbook of Pattern Recognition and Computer Vision*, C.H. Chen, L.F. Pau and P.S.P. Wang editors., pp. 311-368. World Scientific Publishing Company, 1993.

[46] A. Moghaddamzadeh, D. Goldman, and N. Bourbakis. "A Fuzzy-like Approach for Smoothing and Edge Detection in Color Images," *International Journal of Pattern Recognition and Artificial Intelligence*, Vol. 12, No. 6, pp. 801-816, 1998.