

Traffic\_signal\_regognition

Submitted by:

Sourov Sahoo

**ACKNOWLEDGMENT**

**Guidance personals:-**

1. Fliprobo [**Tushar Saraswat**](https://www.flipnwork.com/index.php/team_members/view/16).

**Literature study:-**

<https://www.youtube.com/watch?v=qahpZkPlTRM>

<https://www.youtube.com/watch?v=9jA0KjS7V_c&list=PLZoTAELRMXVPGU70ZGsckrMdr0FteeRUi>

**INTRODUCTION**

* Business Problem Framing

**There are several different types of traffic signs like:---**

speed limits, no entry, turn left or right, children crossing,

no passing of heavy vehicles, etc.

Classify the Traffic signs and identify which class a traffic sign belongs to.

**Solution is**:-

In this project, you have to build a Deep Neural Network model that can classify traffic signs present in the image, into different categories.

With this model,

We should be able to read and understand traffic signs which are a very important task for all autonomous vehicles.

Conceptual Background of the Domain Problem

The images are not very clear which is making it difficult to recognize the exact traffic sign.

* Review of Literature

Data is imbalanced, so categories have very few images whereas other categories have many images.

* Motivation for the Problem Undertaken

Classifying the images of 43 different categories is itself is a challenge.Classifying each image and randomly testing the images is my main motivation behind doing this project.

Moreover,

The images are not very clear which is making it difficult to recognize the exact traffic sign,by tensorflow library we are going to classify the unclear images by resize and array conversion technique with a good accuracy.

**Analytical Problem Framing**

* Mathematical/ Analytical Modelling of the Problem

1. Filtering and resizing(30,30) and converting to array.

Above is the data as per our requirement for feeding into the model for better results.

1. Then appending the data and labels into the empty list we created above.
2. Finally changing both the lists to arrays which is the requirement of model feeding data

4)Train test split with test\_size=20%

5)One-hot-encoding--labels(y—target variable).

* Data Sources and their formats

Primarily data was in .zip file format with no missing values in the data set.

The dataset contains more than 50,000 images of different traffic signs.

It is further classified into 43 different classes.

The dataset is quite varying, some of the classes have many images while some classes have few images. (imbalanced dataset)

The dataset has a train folder which contains images inside each class and a test folder which we will use for testing our model.

The ‘train’ folder contains 43 folders each representing a different class. The range of the folder is from 0 to 42.

Image Data Pre-processing Done

**Images Preprocess technique**

1. Making two empty list to store our dependent and independent features data

#data = [] # dependent features(x) 🡪resized image array

#labels = [] #Independent features(y) -category of the image which it belongs to.

1. Filtering and resize(30,30) and converting to array.

Above is the data as per our requirement for feeding into the model for better results.

c) Then appending the data and labels into the empty list we created above.

d) Finally changing the lists to arrays which is the requirement of model feeding data

* Data Inputs- Logic- Output Relationships

The data input is images resized and converted to array.

The output is the label of the image.

The model is 94% accuracy for test data images.

model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])

* State the set of assumptions (if any) related to the problem under consideration

No Assumptions made .

Hardware and Software Requirements and Tools Used

**Hardware**-64bit, 12GB RAM, 240GB SSD.

**Software-**Excel, Anaconda,jupyter notebook,python 3.6

Libraries used:-

1. numpy

2. pandas

3. matplotlib

4. tensorflow

5. sklearn

6. cv2

7.PIL

8.model.save(‘'output\_trafficsignals.h5')

9. model = load\_model('output\_trafficsignals.h5')

(tensorflow library used to load)

**Model/s Development and Evaluation**

* Identification of possible problem-solving approaches (methods)

Data balancing technique may be used for better results.

* Testing of Identified Approaches (Algorithms)

DL-NN-Model building

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))

# We have 43 classes that's why we have defined 43 in the dense

#The output must be from 0-42 classes present in the dataset.

model.add(Dense(43, activation='softmax'))

#softmax is used in case of image classification problems.

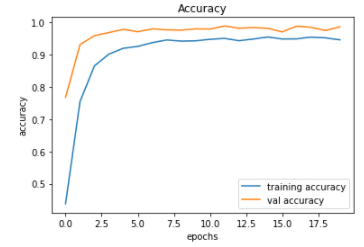
**Model compilation:-**

model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])

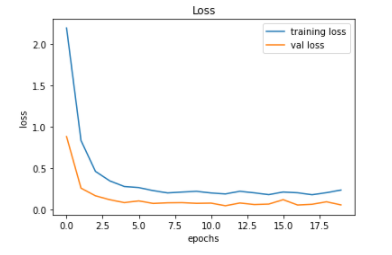
* Key Metrics for success in solving problem under consideration.
* We are choosing **accuracy** as our evaluation metrics for this image classification project.

**Visualizations**

**Training data Accuracy:-**



**Training data Loss:-**



* Interpretation of the Results

We can see on testing , the val accuracy is higher than training accuracy and

loss is less in case of val data which is a very good sign for our model performance.

**Pre-processing:-**

1) Import the required libraries.

2) Import the dataset by zip method extractall( )

Function.

3) Read the train and test dataset.

4) Images Preprocess technique

a) Making two empty list to store our dependent and independent features data

#data = [] #dependent features(x) 🡪resized image array

#labels = [] #Independent features(y) -category of the image which it belongs to.

b) Filtering and resizing(30,30) and converting to array.

Above is the data as per our requirement for feeding into the model for better results.

c) Then appending the data and labels into the empty list we created above.

d) Finally changing the lists to arrays which is the requirement of model feeding data

5) Train test split with test\_size=20%

6) One-hot-encoding--labels(y—target variable).

7) DL-NN model building.

8) Model compilation.

9) Fitting the X\_train , y\_train data for 20 epochs

10) Testing on test data

# Same preprocess techniques used as in train data above

11) Finally Predicting on the images and getting the predicted labels.

12) Checking the Accuracy of the model output for test data.

13) Saving the final model ‘ output\_trafficsignals.h5’

14) Loading the .h5 model

15) Visualization:----------

# Making a dictionary of all 43 Classes of trafic signs.

#Making a function to call the test data (X\_test) and predict(Y\_pred)

#Then just for visualization of different image,just change the image number.

16) Conclusion

**CONCLUSION**

**In this Deep Neural Network model we can classify traffic signs present in the image , into different traffic sign categories.**

**With this model,**

**We are to read and understand traffic signs which are a very important task for all autonomous vehicles.**

* Key Findings and Conclusions of the Study

Imbalanced dataset

* Learning Outcomes of the Study in respect of Data Science

Even bad quality images can be recognized and classified.

* **Limitations of this work. And Scope for Future Work**

Computational complexity:

GridsearchCV is giving a bug related to sklearn.

A way around has to be found for image GridsearchCV.

Cross\_val\_score takes too much time.

Hardware problem:-

Need more powerful system.

My maximum time went in GridsearchCV and model building which is just opposite for a data scientist working hours.

With a upgraded system next time I will be concentrating and spending more time in EDA and data analysis.

**Problems I faced during project**

Unbalanced data set.

GridsearchCV is giving a bug related to sklearn.

A way around has to be found for image GridsearchCV.

I could not play around the values while hypertuning due to low computational power.

Maximum time went in computing rather than analysis of the data.

**Future works**

Try implementing gridsearchcv and

Try different combinations for GridsearchCV hypertuning.