Journal of Theoretical and Applied Information Technology
31st January 2025. Vol.103. No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

UNLOCKING THE POWER OF TRAFFIC SIGN RECOGNITION –AN INTELLIGENT SYSTEM

[1]N. Sudhakar Reddy, [2]Panthangi Venkateswara Rao, [3]Dr. P. Punitha, [4]Venkata Raghu Veeramachaneni, [5]A. S. Malleswari, [6]Anjaneyulu Gurram, [7]Chetla Chandra Mohan

[1]Asst Prof, Dept. of CSE, PVP Siddhartha Institute of Technology Vijayawada, Andhra Pradesh
[2]Asst Prof, Dept. of CSE-VNR Vignana Jyothi Institute of Engineering and Technology, Hyderabad,
[3]Assoc Professor, Dept. of CSE (Data Science), Vignana Bharathi Institute of Technology, Hyderabad
[4]Software Engineer HCL Global Systems INC, USA
[5]Asst Professor Dept. of CSE, Aditya University, Surampalem, Andhra Pradesh
[6]Asst. Professor, Dept. of CSE, Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India.
[1]Asst Prof, Dept. of IT, PVP Siddhartha Institute of Technology Vijayawada, Andhra Pradesh

Email: sudhakar.2215@gmail.com

Abstract

Intelligent transportation systems have become increasingly important in recent years as a means of improving road safety and traffic flow. One key component of these systems is traffic sign recognition, which enables vehicles to automatically detect and interpret traffic signs in real-time. In this paper, we explore the use of advanced machine learning techniques, specifically Convolutional Neural Networks (CNN) and Decision Tree algorithms, for traffic sign recognition. We propose a system that consists of a camera-based traffic sign detection module, a CNN-based recognition module, and a user interface for displaying the detected traffic signs. The system is trained and tested on a large dataset of traffic signs, and we evaluate its performance using a range of metrics, including accuracy, precision, and recall. Our results demonstrate that the proposed system accuracy. We also investigate the impact of various factors, including lighting conditions, weather conditions, and the distance between the camera and the traffic sign. Our results indicate that the proposed system is robust to changes in lighting and weather conditions, and can accurately detect and recognize traffic signs from a range of distances. Overall, this study provides valuable insights into the use of advanced machine learning techniques for intelligent transportation systems, specifically for traffic sign recognition. The proposed system has the potential to significantly improve road safety and traffic flow, and our findings suggest that future research in this area could lead to even more accurate and reliable systems.

Keywords: *Intelligent, Traffic, Sign, Recognition, CNN*

1. INTRODUCTION

Intelligent transportation systems (ITS) are becoming increasingly important in modern transportation infrastructure. These systems utilize advanced technology to improve traffic flow, reduce congestion, and enhance road safety. One of the key components of ITS is traffic sign recognition, which enables vehicles to automatically detect and interpret **traffic** signs in real- time. Traffic sign recognition has the potential to significantly improve road safety and traffic flow by providing drivers with real-time information on speed limits, stop signs, and other important traffic signals. Traditional methods of traffic sign recognition rely on image processing and feature extraction techniques, which often require significant computational resources and may not be robust to changes in lighting and weather conditions. In recent years, machine learning techniques such as Convolutional Neural Networks (CNNs) and Decision Tree algorithms have shown great promise in improving the accuracy and efficiency of traffic sign recognition systems. The goal is to explore the use of CNNs and Decision Tree algorithms for traffic sign recognition in ITS. Specifically, we propose a system that consists of a traffic sign detection module, a CNN- based recognition module which displays the detected traffic signs. We train and test the system on a large dataset of traffic signs, and evaluate its performance using a range of metrics, including accuracy and other factors. Overall, the use of CNNs and Decision Tree algorithms for traffic sign recognition has the potential to significantly improve the accuracy and efficiency of ITS. This

paper provides valuable insights into the development and application of these techniques for traffic sign recognition, with the aim of improving road safety and traffic flow.

2.LITERATURE REVIEW

Alexander et'al study proposes a real-time traffic sign recognition system using convolutional neural networks (CNNs). The proposed system uses a pre-trained CNN model, which is fine-tuned using a large traffic sign dataset. The system achieves high accuracy in traffic sign recognition and shows potential for use in real-world applications.

Giridharan et'al study proposes a traffic flow prediction system using Decision Tree techniques. The proposed system trained traffic data to predict traffic sign. The system achieves accuracy in traffic sign prediction and shows potential for use in ITS applications, such as traffic management and route optimization.

3.PROBLEM IDENTIFICATION AND OBJECTIVES

3.1problem Identification

Intelligent transportation systems (ITS), problem identification may involve identifying issues related to traffic congestion, road safety, public transportation, or environmental concerns. This process typically involves analyzing data and conducting research, to determine the root causes of the problem and identify potential solutions. Once the problem has been identified, ITS technologies can be used to develop solutions and improve the efficiency, safety, and sustainability of transportation systems. The ultimate goal of problem identification is to improve the overall efficiency, safety, and sustainability of transportation systems

3.2objective:

The objective of this study is to develop and evaluate a system for intelligent transportation thatutilizes both Convolutional Neural Network (CNN) and Decision Tree algorithms to recognize and classify traffic signs in real-time. Specifically, it aims to: Develop a CNN model for detecting and localizing traffic signs in images. Implement DecisionTree algorithms for classification of the detected traffic signs into different categories, such as speed limit, stop signs, and pedestrian crossings. Evaluate the performance of the system in terms of accuracy, speed, and robustness under different lighting and weather conditions.By achieving these objectives, it aims to contribute to the development of more advanced and reliable intelligent transportation systems that can enhance road safety and reduce traffic congestion.

4.SYSTEM METHODOLOGY

4.1data Analysis And Pre-Processing

For data analysis and pre-processing, the code reads in image data using pickle and uses libraries such as NumPy and Matplotlib to visualize and preprocess the data. The image data is shuffled, andgrayscale images are created and normalized. For the CNN, the code imports necessary libraries and builds a CNN model using Keras. The model is then compiled and trained using the preprocessed data. For the Decision Tree algorithm, the code imports necessary libraries and loads the data using pickle. The data is then shuffled and split into training and testing sets. A Decision Tree classifier is then instantiated and trained using the training data. Finally, the accuracy of the model is evaluated on the test data.

4.1.1dataset Used

The dataset being used is related to traffic sign classification. The dataset consists of labeled images of traffic signs that are used to train a machine learning model to classify new traffic sign images. The dataset is split into three subsets: training, validation, and testing sets. The training set is used to train the model, the validation set is used to optimize hyper parameters and prevent overfitting, and the testing set is used to evaluate the performance of the model. The dataset contains 43 classes of traffic signs, such as speed limit signs, yield signs, stop signs, and more.

Table1.Data Set used

S.No	Traffic Sign
0.	Limit of speed (20km/h)
1.	Limit of speed (30km/h)
2.	Limit of speed (50km/h)
3	Limit of speed (60km/h)
4	Limit of speed (70km/h)

5	Limit of speed (80km/h)
6	No more speed limit than (80km/h)
7	Limit of speed (100km/h)
8	Limit of speed (120km/h)
9	Passing is not allowed
10	Passing is not allowed for vehicles over 3.5 metric tons
11	Immediately following the intersection, there will be right-of-way
12	Road of Priority
13	Traffic Sign - Yield
14	Traffic Sign - Stop
15	Vehicles are not allowed
16	Prohibition of vehicles over 3.5 metric tons
17	No entry for vehicles
18	Caution: General
19	To the left there is a dangerous curve
20	To the right there is a dangerous curve
21	Curves on both sides
22	Road is Bumpy
23	Road is Slippery
24	On the right, the road narrows
25	Work on Road
26	Signals used by Traffic
27	Pedestrians area
28	Children crossing area
29	Bicycles crossing area
30	Caution: ice and snow area
31	Wild animals crossing area
32	End of passing limits and speed limit.
33	Ahead, make a right turn
34	Ahead, make a left turn
35	Ahead only
36	You can go straight or right
37	You can go straight or left
38	Keep to the right
39	Keep to the left
40	Mandatory Roundabouts
41	No more passing at the end
42	No more passing of vehicles over 3.5 metric tons

4.2model Building

Convolutional Neural Networks (CNNs) for an Intelligent Transportation System (ITS), we can use the following steps:

4.2.1preprocessing

Preprocessing the dataset is an important step in building any machine learning model. In the caseof CNNs, we typically resize the images to a specific size and convert them to grayscale or RGB, depending on the requirements of the model.

4.2.2building The Cnn Model

We can build a CNN model using various layers such as convolutional layers, pooling layers, and fully connected layers. Convolutional layers extract features from the images, pooling layers reduce the size of the feature maps, and fully connected layers classify the images. We can also use techniques like regularization, dropout, and batch normalization to improve the performance of the model.

4.2.3training The Model

We train the CNN model using the preprocessed dataset. We typically use an optimizer such as stochastic gradient descent (SGD) or Adam, and a loss function such as categorical cross-entropy. We train the model for several epochs, adjusting the hyper parameters as necessary, until the validation accuracy reaches a satisfactory level.

Decision Tree for an Intelligent Transportation System (ITS), we can use the following steps:

4.2.4Preprocessing

As with CNNs, preprocessing is an important step in building any machine learning model. In thecase of decision trees, we typically convert the categorical variables to numerical variables and handle any missing values in the dataset.

4.2.4building The Decision Tree Model

Decision trees are built by recursively splitting the dataset based on a set of rules. At each node,
we select the feature that maximally separates the data based on a criterion such as Gini impurity or entropy. We can also use techniques like pruning to prevent overfitting.

4.2.5training The Model

We train the decision tree model using the preprocessed dataset. We typically split the dataset into a training set and a validation set, and use

techniques like cross-validation to tune the hyper parameters of the model.

4.3model Training

For Intelligent Transportation System (ITS), we can use CNN and Decision tree algorithms for model training. In CNN, we can train the model using the dataset to get the accuracy of a correct prediction. We can set the number of epochs to train the model and then calculate the training- validation loss and accuracy curves to examine the model. Similarly, for Decision tree algorithm,we can train the model using the dataset and calculate the accuracy of the model. The aim is to getthe highest accuracy and the loss value is also computed along with the accuracy to show how much loss occurred during training.

4.3.1model Evaluation

A machine learning model's performance, as well as its advantages and disadvantages, are understood through the process of model evaluation, which employs various evaluation metrics. Model effectiveness must be evaluated in the early stages of research, and model evaluation also aids in model monitoring.

5.OVERVIEW OF TECHNOLOGIES

5.1machine LEARNING

Machine learning (ML) is a branch of Artificial Intelligence (AI) that enables machines to automatically learn from data and prior experiences to recognize trends and make predictions with minimal human supervision. Machine learning is concerned with using data and algorithms to mimic how individuals learn, continually improving its accuracy. Machine learning performs tasks without explicit instructions. Machine learning analyzes algorithms and structures within a dataset and attributes of that dataset are used as inputs to the algorithms during training and testing. The accuracy of the input-data representation is often a key factor in determining how well a machine learning algorithm performs. It has been demonstrated that a good data representation outperforms a bad one in terms of performance. Machine learning is now widely employed in research and has a wide range of uses, such as picture classification.

5.2 Convolutional Neural Networks

Convolutional Neural Networks (ConvNet) are a particular kind of neural network with a focus on handling data having a grid-like layout, like photographs. Digital images are a form of binary visual data. Pixel values, which specify how bright and what hue each pixel should appear, are arranged in a grid-like pattern. CNN does not require any preprocessing and can run directly on an underdone image. Convolutional Neural Networks are popularly known for image analysis, but they've also been employed in other areas of machine learning, such as natural language processing. In order to collect precise data about the visual field, CNNs work by layering filters along each dimension (horizontal or vertical). Each of the many convolutional layers that make up CNN is able to recognize increasingly complex structures. By developing a neural network that operated similarly to the human brain in 1982, Fukushima began work on visual cortex modeling. He did this by taking into account the two ideas listed below. Neurons grow continuously over the visual field. Information from simple cells is combined by complex cells (orientation-selective units). As a result, altering the image will change how simple cells activate, but not how complex cells activate in concert (convolutional pooling). In CNN, a small input window can be moved across
the full image. A particular pattern is found when the input window is activated. Based on Fukushima's research on visual cortex modeling (simple/complex cell hierarchy), Yan Lecun trained the CNN for handwritten digit recognition using back propagation in 1990.

5.2.1Applications of Convolutional Neural Networks

Convolutional neural networks (CNNs) perform admirably in many computer vision tasks. Here are some examples of CNN in action:

Self-Driving Cars

In the context of automatic vehicles, CNN has been used to recognize obstacles and decipher traffic signs. In order to enhance how CNN models react to certain situations, reinforcement learning—a subfield of machine learning that focuses on both positive and negative feedback from the environment—has been used in conjunction with CNNs.

Document Classification

Documents have been categorised using CNNs. CNN models may identify a document's category based on its content, such as news articles about politics or sports. By locating words and phrases associated with the subject matter of a given document, CNN networks can use text and visuals to understand what is contained inside. CNN models have also been used to document

summaries based on their content, highlighting and elaborating on important ideas.

Face Detection

CNNs have been used to recognize faces in pictures. The network uses an image as input and outputs a set of values that represent different aspects of faces or facial traits. According to numerous studies, CNN successfully recognizes faces 97% of the time, which is a higher accuracy rate than earlier algorithms. CNN has the potential to lessen face deformation as well. CNNs can accurately identify facial features like the eyes, nose, and mouth, minimizing distortions brought on by angles or shadows.

Applications like text classification and biometric authentication have employed CNN in the real world. CNN networks analyses an image as input and generate responses in natural language for automatic translation or visual question-answering applications. By highlighting specific portions in documents, CNN has even been able to summarize them.

5.3 Decision Tree Classifier

A decision tree classifier is a machine learning algorithm that builds a model in the form of a tree structure to classify data. It is a type of supervised learning algorithm used for classification tasks where the data is labeled.

The decision tree classifier starts at the root node and makes a series of binary splits based on the feature values of the data. The decision tree splits the data into branches, with each branch representing a possible outcome or decision based on the value of the selected feature. This process continues recursively until each leaf node represents a class label or a decision.

In a decision tree, each internal node represents a test on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label. The goal of the algorithm is to create a tree that predicts the class labels of the input data as accurately as possible.

Decision trees are popular because they are easy to interpret and visualize. They are also non-parametric, meaning they do not make any assumptions about the underlying distribution of the data. However, decision trees can suffer from overfitting if they are not pruned or if they are too complex, which can lead to poor generalization on new data.

5.3.1Application of Decision Tree

Credit risk assessment

Decision tree classifiers can be used to evaluate the creditworthiness of individuals or businesses by analyzing their credit history, financial statements, and other relevant factors.

Medical diagnosis

Decision tree classifiers can be used to help diagnose medical conditions by analyzing patient symptoms, medical history, and other clinical data.

Fraud detection

Decision tree classifiers can be used to detect fraudulent activities in financial transactions by analyzing various attributes such as transaction amount, location, and frequency.

Customer segmentation

Decision tree classifiers can be used to segment customers into different groups based on their behavior, demographics, and other characteristics. This information can be used for targeted marketing and improving customer retention.

Predictive maintenance

Decision tree classifiers can be used to predict equipment failure or maintenance needs by analyzing historical data on equipment performance and other factors.

Churn prediction

Decision tree classifiers can be used to predict customer churn or attrition by analyzing customer behavior, preferences, and other factors. This information can be used to develop targeted retention strategies.

5.4 Packages, Libraries And FunctioNS

Package/Library	Use
matplotlib	It is a large Python library for creating static,animated, and interactive visualizations.
numpy	The Python library for performing mathematicaland logical operations on arrays.
pandas	It is an open source tool for data processing and analysis that is speedy, powerful, versatile, and easy to use.
drive	It is used to import data from Google Drive
tensorflow	TensorFlow is used to implement best practicesfor model training, performance monitoring andmodel tracking.

keras	It is a neural network API for python with tensorflow which is used to build machine learning models.
pickle	It is used for serialization and de-serialization ofPython objects, which means converting objectsinto a stream of bytes and vice versa.
random	It is used to pick random numbers, letters,images, etc.
sklearn	It provides a wide range of tools for tasks like data preprocessing, feature extraction, model selection, and evaluation.
confusion matrix	It shows the number of true positives, false positives, true negatives, and false negatives foreach class in the dataset.

6.IMPLEMENTATION

6.1CODING

(a) Using CNN

1. Import all Libraries to be used in this project and mounting the drive.

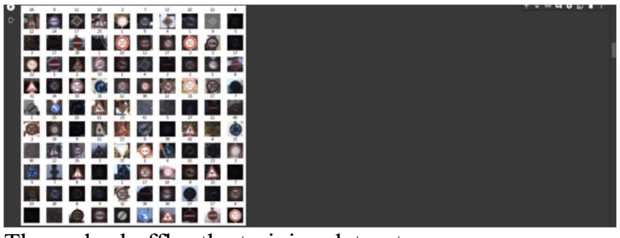
The code loads the training, validation, and test datasets using pickle.

It extracts the features and labels from the dataset.

Print various shapes of the training, testing and validation dataset.

The code displays a random image from the training set.

It creates a grid of random images from the training set using matplotlib.

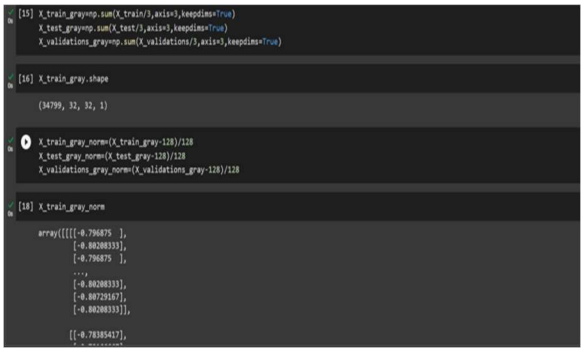


The code shuffles the training dataset.

It converts the color images to grayscale and normalizes them.

Journal of Theoretical and Applied Information Technology
31st January 2025. Vol.103. No.2

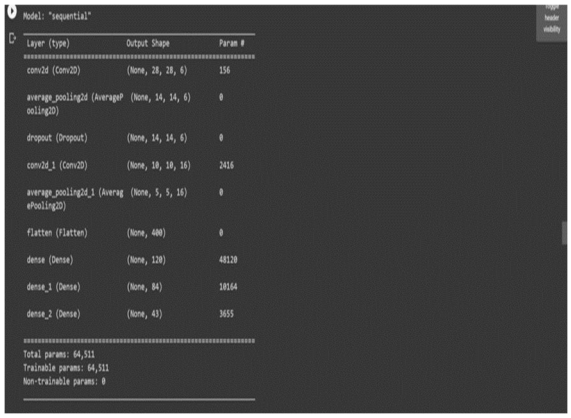
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



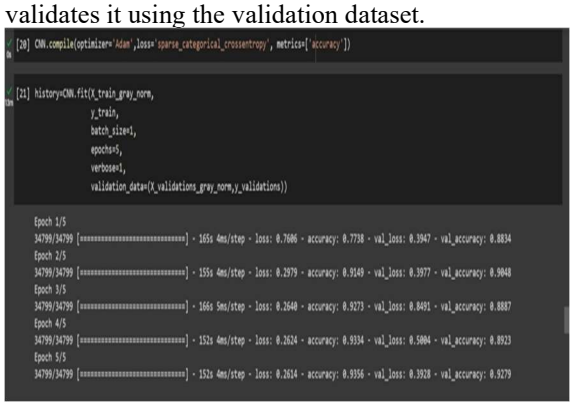
The code displays a random grayscale image from the training set.

It builds a convolutional neural network (CNN) using TensorFlow's Keras API.

The CNN comprises two convolutional layers, one dropout layer, and three dense layers.



The code compiles the CNN using the Adam optimizer and sparse categorical cross-entropy loss and it trains the CNN using the training dataset and validates it using the validation dataset.

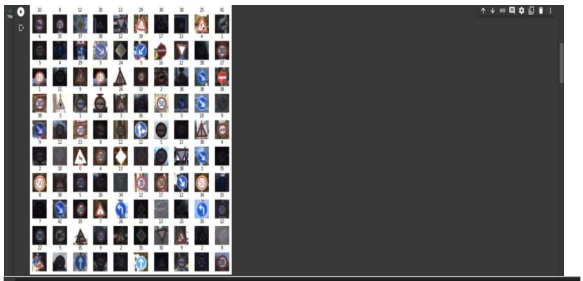


The code evaluates the CNN's performance on the test dataset and prints the test accuracy.

It visualizes the training and validation loss and accuracy using matplotlib

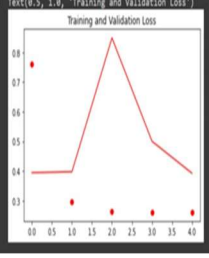
Journal of Theoretical and Applied Information Technology
31st January 2025. Vol.103. No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



```
epochs = range(len(accuracy))
plt.plot(epochs, loss, 'ro', label = 'Training Loss')
plt.plot(epochs, val_loss, 'r', label = 'Validation Loss')
plt.title('Training and Validation Loss')
```

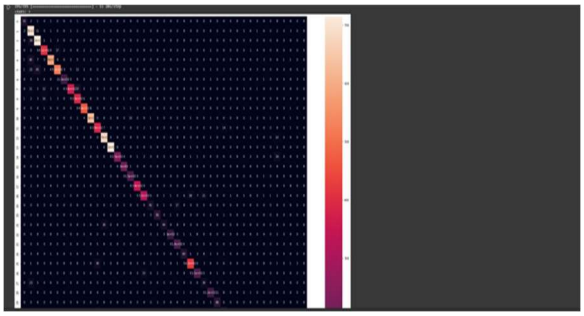
```
Text(0.5, 1.0, 'Training and Validation Loss')
```



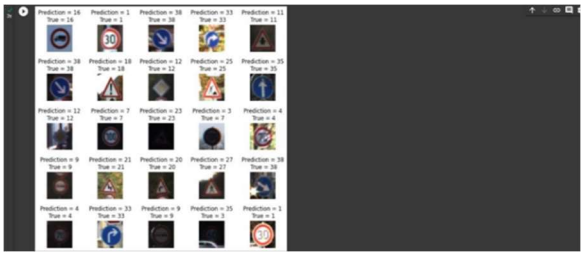
The code predicts the class labels for the test dataset using the trained CNN and it computes and visualizes the confusion matrix for the test dataset.

```
predicted_classes = np.argmax(CNN.predict(X_test_gray_norm), axis=1)
y_true = y_test

from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_true, predicted_classes)
plt.figure(figsize = (25,25))
sns.heatmap(cm, annot = True)
```



The code displays a grid of random images from the test set with their predicted and trueclass labels.



(a) Using Decision Tree Algorithm

Import necessary libraries.

```
[4] import matplotlib.pyplot as plt
    import numpy as np
    import pandas as pd
    import seaborn as sns
    import pickle
    import random

    from sklearn.tree import DecisionTreeClassifier
    from sklearn.metrics import accuracy_score
    from sklearn.model_selection import train_test_split
    from sklearn.utils import shuffle
```

Create a 10x10 grid of subplots using the plt.subplots() method and plot 100 random imagesfrom the training dataset.

```
# Visualize some images
w_grid = 10
l_grid = 10
fig, axes = plt.subplots(l_grid, w_grid, figsize = (10,10))
axes = axes.ravel() # Flatten the 5 x 5 matrix into 25 array
n_training = len(X_train) # get the length of the training dataset
for i in np.arange(0,w_grid*l_grid):
    index=np.random.randint(0,n_training)
    axes[i].imshow(X_train[index])
    axes[i].set_title(y_train[index],fontsize=10)
    axes[i].axis('off')
plt.subplots_adjust(hspace=0.4)
```

Shuffle the training data using the shuffle() method from sklearn.utils.

Split the training data into 80% training set and 20% validation set using train_test_split() method from sklearn.model_selection.

Create an instance of the DecisionTreeClassifier model with the random_state parameter set to 0 and train the model on the training set using the fit() method.

Predict the labels of the training and validation sets using the predict() method and calculate the accuracy score using the accuracy_score() method.

Load the testing dataset and extract the features and labels from it.

Preprocess the test data by converting RGB images to grayscale and normalizing the pixel values.

Reshape the training and testing data into 2D arrays using the reshape() method and the max_depth parameter of the DecisionTreeClassifier model is set to 5.

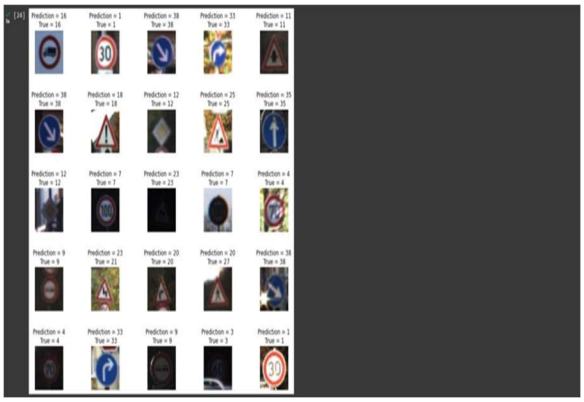
Train the model on the training set using the fit() method and evaluate the model's performanceon the test set using the score() method.

Predict the labels of the test set using the predict () method and calculate the accuracy score using the accuracy_score () method.

Create a 5x5 grid of subplots and plot 25 random images from the test set along with their predicted and true labels.

Journal of Theoretical and Applied Information Technology
31st January 2025. Vol.103. No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



6.TESTING

After completing the coding part, we now compare the output of the techniques that we have usedin the coding part.

Table 2: Comparison of Technique , Epoch, Loss and Accuracy

S.No.	Technique	Accuracy
1.	CNN	93.61%
2.	Decision Tree	94.63%

7.RESULTS

The CNN model was trained for 5 epochs using the training set. The loss and accuracy were computed to investigate the model. Our model was 93.61% accurate.

Other model used is Decision Tree which was trained. The accuracy were computed to investigate the model. Our model was 94.63% accurate.

8.CONCLUSION AND FUTURE SCOPE

Intelligent transportation systems (ITS) have become an important area of research and development in recent years, with traffic sign recognition being a crucial component of these systems. With the help of computer vision and machine learning, traffic sign recognition systems can detect and interpret road signs in real-time, providing drivers with critical information about speed limits, directions, and other traffic regulations.

The advantages of traffic sign recognition systems are numerous. They can reduce the cognitive burden on drivers, improve situational awareness, and enhance road safety. Additionally, these systems can help reduce traffic congestion, improve traffic flow, and reduce the environmental impact of transportation.

Despite its many benefits, traffic sign recognition is still an evolving technology, and there are challenges that need to be addressed, such as dealing with adverse weather conditions and the development of standardized signage. However, with ongoing research and development, traffic sign recognition is expected to become an increasingly important technology in modern transportation systems.

9.FUTURE SCOPE

The future scope of traffic sign recognition is vast, as advancements in technology and research continue to evolve the capabilities of intelligent transportation systems. As the development of autonomous vehicles progresses, traffic sign recognition will become an even more critical component of these vehicles' navigation systems. By leveraging advanced machine learning algorithms and sensors traffic sign recognition systems can provide autonomous vehicles with critical information about road conditions, obstacles, and traffic regulations. Additionally, future traffic sign recognition systems may incorporate vehicle-to-infrastructure (V2I) communication to improve the accuracy and timeliness of traffic sign information. As transportation technology continues to advance, traffic sign recognition is likely to play an increasingly important role in enhancing safety, efficiency, and sustainability on our roads.

REFERENCES

[1]“Mallik, Sumit. "Intelligent transportation system." International Journal of Civil Engineering Research 5.4 (2014): 367-372.

[2]Jarašūniene, Aldona. "Analysis of possibilities and proposals of intelligent transport system (ITS) implementation in Lithuania." Transport 21.4 (2006): 245-251.MILES, C. J.; CHEN, K. The intelligent transport system, UK 2004.

[3]Regan, Michael A., et al. "Acceptability of in-vehicle intelligent transport systems to Victorian car drivers." Monash University Accident Research Centre (2002).

[4]Rodrigue, Jean-Paul. The geography of transport systems. Routledge, 2020.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[5]Little, Cheryl. "The intelligent vehicle initiative: advancing" human-centered" vehicles." Public Roads 61.2 (1997): 18-25.Tan, Y. Wu, B. Shen, P. J. Jin and B. Ran, "Short-Term Traffic Prediction Based Dynamic Tensor Completion".

[6]Yazdizadeh, Ali, and Bilal Farooq. "Smart mobility ontology: Current trends and future directions." Handbook of smart cities (2020): 1-36.

[7]Niranjan, S. K., et al. "Smart Technology for Smart Nation SmartTechCon2017."

[8] Ghosh, Raksha, et al. "Intelligent transportation systems: A survey." 2017 International Conference on Circuits, Controls, and Communications (CCUBE). IEEE, 2017.