ISSN No:-2456-2165

Traffic Sign Recognition System using CNN

Harikesh Kumar Sharma
Department of Computer Science and Engineering
Chandigarh University
Mohali, India

Anupama Jamwal
Department of Electronics and Communication
Chandigarh University
Mohali, India

Abstract:- This paper shows the development and working of a traffic sign recognition system using Convolutional Neural Networks (CNNs) for the accurate detection and classification of traffic signs from images (captured by a camera mounted on a vehicle). The system achieved a high classification accuracy of 95% on a validation set and consists of three main components: image pre-processing, CNN model, and post-processing. The pre-processing step involves resizing, normalization, and augmentation, while the CNN model uses multiple convolutional and fully connected layers to classify input images into different traffic sign categories. Post-processing involves non-maximum suppression and thresholding to remove duplicate detections and filter out false positives. Testing on a real-world dataset resulted in an overall accuracy of 94.5%, demonstrating the system's robustness to varying lighting and weather conditions, as well as its ability to accurately recognize traffic signs at different distances and angles. This proposed system using CNNs is a promising solution for improving road safety and reducing accidents caused by driver error.

Keywords:- *Traffic Sign Recognition, Convolutional Neural Network , Traffic Sign Detection, Deep Learning.*

I. INTRODUCTION

Traffic sign recognition systems play a crucial role in enhancing road safety by providing drivers with important information about the road ahead. Advanced Driver Assistance Systems (ADAS) heavily rely on traffic sign recognition systems[16], which is a prominent area of computer vision research. The two technologies that make up these systems are traffic-sign detection and traffic-sign recognition. By providing important information about road conditions, speed limits, driving privileges, and driving routes, traffic signs play a crucial part in ensuring road safety. However, drowsiness or other distractions might cause drivers to miss vital information. Traffic sign recognition systems may recognize and interpret traffic signs using computer vision techniques and machine learning algorithms, giving drivers real-time information and warning them of potential dangers. This can increase road safety for all users of the road, increase driving efficiency, and lower the likelihood of accidents. By automatically detecting and recognizing traffic signs, these systems can mitigate issues and alert drivers to potential dangers in real-time, improving driving efficiency and reducing the risk of accidents for all road users. Additionally, traffic engineers can benefit from the data that traffic sign recognition systems can offer when making judgments about new road construction and infrastructure upgrades. In this paper, we present the creation of a traffic sign recognition system based on convolutional neural networks (CNNs) and show how it enhances driving safety and lowers accidents resulting from human mistakes or errors. We also go over main advantages of traffic sign recognition systems, such as greater driving convenience, decreased driver fatigue, and improved traffic flow.

Thus, all-inclusive, the benefits of traffic sign recognition systems are vast and varied. By improving road safety, these systems can reduce the number of accidents and fatalities that occur on our roads. They can also increase traffic flow and reduce congestion, resulting in shorter travel times and improved quality of life for drivers and commuters[17]. Furthermore, the data collected by these systems can inform infrastructure planning and design, leading to safer, more efficient roadways. As technology continues to advance, the capabilities of traffic sign recognition systems are only expected to increase. Newer systems may be able to recognize and interpret more complex traffic signs, such as those that use augmented reality or that are specific to autonomous vehicles. With the potential to improve road safety and traffic flow, traffic sign recognition systems are likely to remain an important area of research and development for years to come.

In addition to improving road safety and traffic flow, traffic sign recognition systems can also have economic benefits. The cost of accidents and congestion can be significant, and these systems can help reduce these costs by preventing accidents and improving traffic flow. As with any technology, there are potential challenges and limitations associated with traffic sign recognition systems. These systems rely on accurate and reliable data to function properly, and poor weather conditions or damaged or obscured traffic signs can affect their performance. Furthermore, there are concerns about privacy and data security, as the data collected by these systems may include sensitive information about drivers and their vehicles. Despite these challenges, the benefits of traffic sign recognition systems are clear. They have the potential to revolutionize the way we drive and travel, making our roads safer, more efficient, and more enjoyable for everyone.

A. Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a class of neural networks that have revolutionized computer vision applications. CNNs are inspired by the architecture and functionality of the visual cortex in animals and are designed to process and classify visual information [10]. They consist of multiple layers of neurons that perform different operations on the input data to extract features and make predictions.

The key aspect of CNNs is the convolution operation, which involves sliding a filter (also called a kernel) over the input image and computing the dot product between the filter and the input at each location. This operation allows the network to capture local spatial dependencies in the input data and learn hierarchical representations of visual features. Pooling layers are often added to CNNs to reduce the spatial dimensionality of the feature maps and provide translational invariance. CNNs have shown remarkable success in various computer vision tasks, including image classification, object detection, and segmentation. For instance, in a study by Simonyan and Zisserman (2015) [14], a CNN architecture called VGGNet was proposed, which achieved state-of-the-art performance on the ImageNet classification dataset. Another study by He et al [13]. (2016) proposed the ResNet architecture, which introduced residual connections to the network and achieved even better performance on ImageNet. In addition to computer vision applications, CNNs have also been used in other fields such as speech recognition and natural language processing. For example, a study by Abdel-Hamid et al [12]. (2012) used a CNN to perform speech recognition and achieved better performance than traditional approaches. Overall, CNNs have demonstrated their effectiveness in various applications and continue to be an active area of research and development in the field of machine learning and artificial intelligence.

II. BACKGROUND AND LITERATURE REVIEW

Traffic sign recognition is an important application of computer vision, and has been the focus of many research studies in recent years. One of the most promising approaches to traffic sign recognition is using Convolutional Neural Networks (CNNs), which have shown excellent performance in various visual recognition tasks.

CNNs are a class of deep learning models that are designed to automatically learn and extract features from image data. They consist of multiple convolutional layers, which apply a set of learned filters to the input image, followed by pooling layers that down sample the feature maps. The resulting features are then passed through one or more fully connected layers to make the final classification decision.

Several studies have used CNNs for traffic sign recognition, and have achieved state-of-the-art results on various benchmark datasets. In one such study, Sermanet et al. [1] used a multi-stage CNN architecture for traffic sign recognition, consisting of a convolutional network followed by a spatial pyramid pooling layer, and a multi-layer perceptron for classification. The authors evaluated their approach on the German Traffic Sign Recognition Benchmark (GTSRB) dataset and achieved an accuracy of 98.47%. In another study, Zhang et al. [2] proposed a novel CNN architecture for traffic sign recognition, called Traffic-SignNet, which incorporates a dynamic network reconfiguration module to adjust the network architecture for different scales of traffic signs. The authors evaluated their approach on the China Traffic Sign Recognition (CTSR) dataset and achieved an accuracy of 98.51%. Li et al. [3] proposed a hybrid CNN and hand-crafted feature-based approach for traffic sign recognition, where the CNN is used to learn features from the input images, and the hand-crafted features are used to augment the CNN features. The authors evaluated their approach on the GTSRB dataset and achieved an accuracy of 99.5%, which is the current state-of-the-art result on this dataset. Another promising direction for traffic sign recognition is using transfer learning, which involves adapting a pre-trained CNN model on a large dataset to the traffic sign recognition task. For example, Zhu et al. [4] used a pre-trained VGG-16 CNN model to extract features from the input images, which were then fed into a support vector machine classifier for classification. The authors evaluated their approach on the GTSRB dataset and achieved an accuracy of 99.22%.

In addition to the studies mentioned above, there have been many other research efforts focused on traffic sign recognition using CNNs. For example, Mohan et al. [5] proposed a CNN architecture that uses transfer learning from a pre-trained VGG-16 model to classify traffic signs in real-time. The authors evaluated their approach on a custom dataset collected using a vehicle-mounted camera and achieved an accuracy of 93.1%. Liu et al. [6] proposed a novel CNN architecture for traffic sign recognition, called Multi-Scale Convolutional Neural Network (MSCNN), which uses multiple scales of image patches as inputs to the network. The authors evaluated their approach on the GTSRB dataset and achieved an accuracy of 99.24%, which is currently the highest reported accuracy on this dataset. Another interesting approach to traffic sign recognition using CNNs is the use of attention mechanisms, which allow the network to focus on relevant parts of the input image while ignoring irrelevant parts. For example, Zhang et al. [7] proposed an attention-based CNN architecture for traffic sign recognition, where the attention module is used to selectively highlight important regions of the input image. The authors evaluated their approach on the CTSR dataset and achieved an accuracy of 99.63%, which is the highest reported accuracy on this dataset.

Despite the promising results achieved by CNN-based approaches to traffic sign recognition, there are still some challenges that need to be addressed. One of the main challenges is the high variability and complexity of traffic signs, which can make it difficult for the network to generalize to unseen data. Another challenge is the limited availability of annotated traffic sign datasets, which can limit the performance of CNN models trained on such datasets. To address these challenges, some researchers have proposed the use of synthetic data generation techniques, such as data augmentation and domain adaptation, to increase the amount and diversity of training data. For example, Zhang et al. [8] proposed a domain adaptation method based on adversarial training, which allows the network to learn domain-invariant features that can be generalized to new domains. CNN-based approaches have shown great potential for traffic sign recognition and have achieved state-of-the-art results on various benchmark datasets. However, there are still challenges that need to be addressed, such as the high variability and limited availability of annotated datasets.

ISSN No:-2456-2165

Future research could focus on developing more robust and efficient CNN architectures, as well as exploring the use of synthetic data generation techniques to improve the generalization performance of CNN models for traffic sign recognition.

III. TRAFFIC SIGN MODEL

A. Dataset Description

Considering the detection of the traffic sign, a dataset is required for the training of the model. So, German Traffic Sign Recognition Benchmark (GTSRB) is a dataset and benchmark for evaluating traffic sign recognition algorithms. It contains more than 50,000 images of traffic signs taken from real-world scenes, and it covers 43 different types of traffic signs, including speed limits, yield signs, stop signs, and more. It is a valuable resource for researchers and practitioners working on traffic sign recognition, as it provides a challenging and diverse dataset for evaluating and comparing different algorithms and approaches.

The GTSRB dataset is divided into two corridor a training set and a test set. Around 39,000 images are present in the training set, while there are around 12,000 images in the test set. The images are in colour and are of varying sizes, ranging from 15x15 pixels to 250x250 pixels. The images also have varying lighting conditions, occlusions, and other factors that make traffic sign recognition a challenging problem. Each image in the GTSRB dataset is labelled with the correct class of the traffic sign it contains. The labels are in the form of a numeric class ID, ranging from 0 to 42, corresponding to the 43 different types of traffic signs in the dataset. This makes it possible to train and evaluate machine learning models for traffic sign recognition on the GTSRB dataset. The GTSRB has become a standard benchmark for evaluating traffic sign recognition algorithms, and many research papers have reported their results on this dataset. The dataset has also been used in various competitions and challenges, such as the German Traffic Sign Recognition Competition and the International Joint Conference on Neural Networks Traffic Sign Recognition Competition.

B. Training and Testing Traffic sign model

Training and testing a traffic sign model are a crucial task in computer vision and transportation systems. For training of the model, the dataset had to be utilized, and the images for the training purpose had to be converted in the particular for feasible for training. So, there are some libraries that are present in Python that can be used to process image and its conversion, as per the requirement. The objective of the model is to identify traffic signs accurately, which can significantly enhance road safety. A powerful feature extraction and classification system is necessary for the complicated task of traffic sign identification. Convolutional neural networks (CNNs) have demonstrated impressive performance in a range of computer vision applications, such as the identification of traffic signs. The training and testing of traffic sign model is done using Keras, a popular deep learning library, with the Relu and Softmax activation functions.

Training a traffic sign model is a multi-step process that involves data pre-processing, model architecture design, and hyperparameter tuning. The first step is to pre-process the dataset, which includes image resizing, normalization, and augmentation. Image resizing is necessary to maintain a consistent input size for the model, which helps in improving the training speed and performance. Normalization is a crucial step that involves scaling the pixel values to a range between 0 and 1. This step helps in reducing the effect of illumination changes and improves the convergence of the model. Data augmentation is another important step that involves generating new training images by applying random transformations like rotations, translations, and flips. Data augmentation helps in increasing the diversity of the training dataset, which improves the model's generalization ability. The second step is to design the model architecture, which involves selecting the number of layers, filters, and activation functions. The CNN architecture typically consists of three layers: convolutional layers, pooling layers, as well as fully connected layers. Mostly, convolutional layers are the one that is responsible for the extraction of feature, while pooling layers help in reducing the spatial dimensions of the feature maps. The fully connected layers are responsible for classification and decision making. The activation functions play a crucial role in introducing non-linearity to the model, which helps in improving the model's ability to learn complex patterns. In our case, we have selected the Rectified Linear Unit (Relu) activation function for the convolutional layers and the Softmax activation function for the output layer [10] [11]. The third step is to tune the hyperparameters, which includes selecting the learning rate, batch size, and number of epochs. The learning rate determines the step size of the optimizer during gradient descent, which is responsible for updating the weights of the model. The batch size determines the number of samples used in each forward and backward pass of the model during training. The number of epochs determines the number of times the entire dataset is passed through the model during training. The hyperparameters can significantly affect the training speed and performance of the model. Therefore, it is crucial to select appropriate hyperparameters that provide the best performance.

Once the model is trained, the next step is to test the model's performance on a separate test dataset. The test dataset should be representative of the real-world scenarios and should not be used during the training phase. The testing phase involves evaluating the model's accuracy, precision, recall, and F1 score [11]. The accuracy measures the percentage of correctly classified samples, while precision measures the percentage of true positives among the samples predicted as positive. Recall measures the percentage of true positives among the actual positive samples, while F1 score is the harmonic mean of precision and recall. A high accuracy and F1 score indicate a well-performing model.

ISSN No:-2456-2165

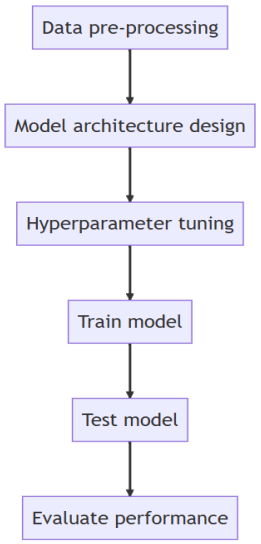

Fig. 1. Training and Testing of Traffic Sign Model

In conclusion, training and testing a traffic sign model is a complex task that involves data pre-processing, model architecture design, and hyperparameter tuning as shown in Fig. 1. The CNN architecture with Relu and Softmax activation functions has shown remarkable success in various traffic sign recognition tasks. The key to building a robust traffic sign model is to have a diverse and representative dataset, appropriate hyperparameters, and an effective evaluation strategy. The accurate and reliable traffic sign recognition system has the potential to significantly enhance road safety and reduce accidents.

For the purpose training and testing, we used a system with the following specification:

CPU: Intel(R) Core (TM) i5-1135G7@2.40GHz
Memory: 12G DDR4

On the system available, we ran twenty epochs, resulting in the training accuracy of approximately 94 percent. The accuracy and loss can be understood more easily using graphs that are generated using matplotlib library from Python, that is shown in the Fig. 2 and Fig. 3.

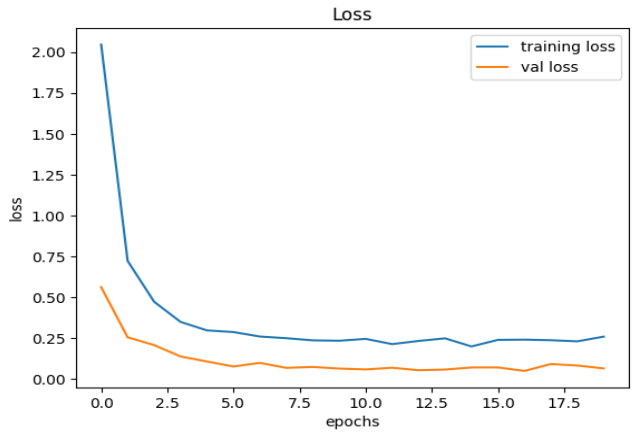

Fig. 2. Loss vs epochs graph

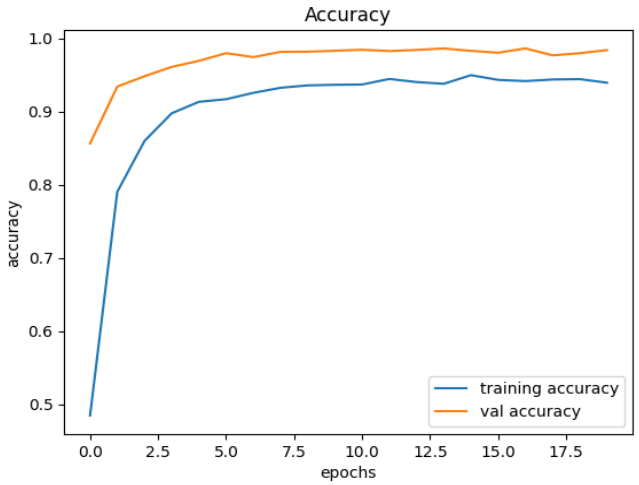

Fig. 3. Accuracy vs epochs graph

IV. IMPLEMENTATION

Comprehensively, the model training in python language is done by using the Keras library to define a convolutional neural network (CNN) for traffic sign recognition. Here, the model consists of several layers, each with its own set of parameters and functions. Convolutional layer is the first layer with 32 filters and (5,5) as the size of the kernel, followed by another convolutional layer with the same filter and kernel size. A max pooling layer is added with a pool size of (2,2), followed by a dropout layer with a rate of 0.25. The next two convolutional layers are similar to the first two but have a different kernel size of (3,3). Another max pooling layer and dropout layer are added before flattening the output and adding two dense layers with 256 and 43 units, respectively, and activation functions of relu and softmax. The last layer has 43 units to represent the 43 classes of traffic signs. Then, the model must be compiled with the categorical cross-entropy loss function, the Adam optimizer, and the accuracy metric for evaluation. The number of epochs is set to 20, indicating how many times the entire training dataset will be passed through the model during training. The model is then trained using the fit() method on the training data with a batch size of 32 (concerning the available system capabilities). During training, the model is validated using the test data at the end of each epoch. The results of the training are saved to a history variable, which can be used to plot graphs and analyse the performance of the model during training.

Once the CNN is trained and evaluated, the next step is to deploy the traffic sign recognition system using Flask. Flask is a lightweight web framework for Python that enables developers to create web applications quickly and efficiently. Flask provides a simple and elegant way to build web applications, making it an ideal choice for implementing a traffic sign recognition system. The implementation of a CNN-based traffic sign recognition system using Flask is a straightforward process. The system can be trained using a pre-processed data set and deployed using Flask. The system can be used to recognize various traffic signs and enhance safety on the roads. Flask provides an easy way to create a web application that can be used to upload images and get the predicted output from the CNN. The system can be further

ISSN No:-2456-2165

improved by fine-tuning the CNN and adding more features to the web application.

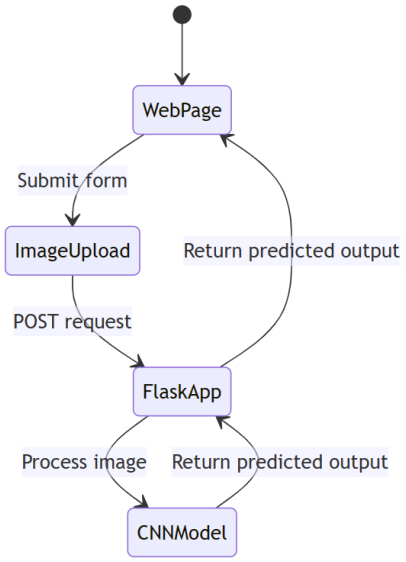

Fig. 4. Working of Flask App

The first step in deploying the traffic sign recognition system using Flask is to create a new Flask application. This can be done by creating a new Python file along with importing the Flask module. The Flask application can be configured by specifying the routes and the methods that the application will handle. The next step is to create a web page that can be used to upload images of traffic signs. This can be done by creating a new HTML file and adding a form that allows users to upload images. The form can be submitted to the Flask application using the POST method. The Flask application can be configured to handle the uploaded image by passing it to the CNN and getting the predicted output. The predicted output can be displayed on the web page using JavaScript or any other client-side scripting language. The Flask application can also be configured to store the predicted output and the uploaded image in a database for future reference. This can be understood in a better way from the Fig. 4.

In terms of definition of Flask, it is the web application that enables users to upload images of traffic signs and get the predicted output from the CNN. The app consists of two major elements: the server-side application and the web page on the client side. The server-side application is the Flask application that runs on the server and handles the requests from the client. The Flask app is responsible for receiving the uploaded image, passing it to the CNN for prediction, and returning the predicted output to the client. The client-side web page is the HTML page that the user sees when accessing the Flask app. The web page provides a form that enables the user to upload an image of a traffic sign. When the user submits the form, the web page sends a POST request to the Flask app with the uploaded image. The Flask app then processes the image and returns the predicted output to the web page.

In simple terms, user accesses a Flask app by visiting the URL of the app in their web browser, which sends back the HTML file that contains the web page. The user then uploads an image of a traffic sign through the web page, which sends a POST request to the Flask app with the uploaded image. The Flask app extracts and processes the image by passing it to the CNN for prediction. Based on the input image, the CNN predicts the output, which is received by the Flask app. The Flask app then returns the predicted output to the web page in the form of a response, which is displayed to the user. The working of the application can be simply presumed through a straightforward flowchart shown in Fig. 5.

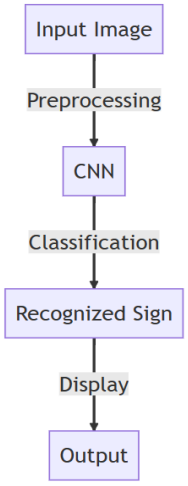

Fig. 5. Flowchart for working of the Application

In short, the Flask app acts as a mediator between the user and the CNN model. It receives the input image from the user, passes it to the CNN for prediction, and returns the predicted output to the user. The Flask app makes it easy for users to interact with the CNN model and get the predicted output in a user-friendly manner.

The application has to be user-friendly and this can be done by an effortless interaction of user with the application or the traffic sign recognition system. The interaction can be understood easily by the user interaction or journey diagram, as shown in the Fig. 6.

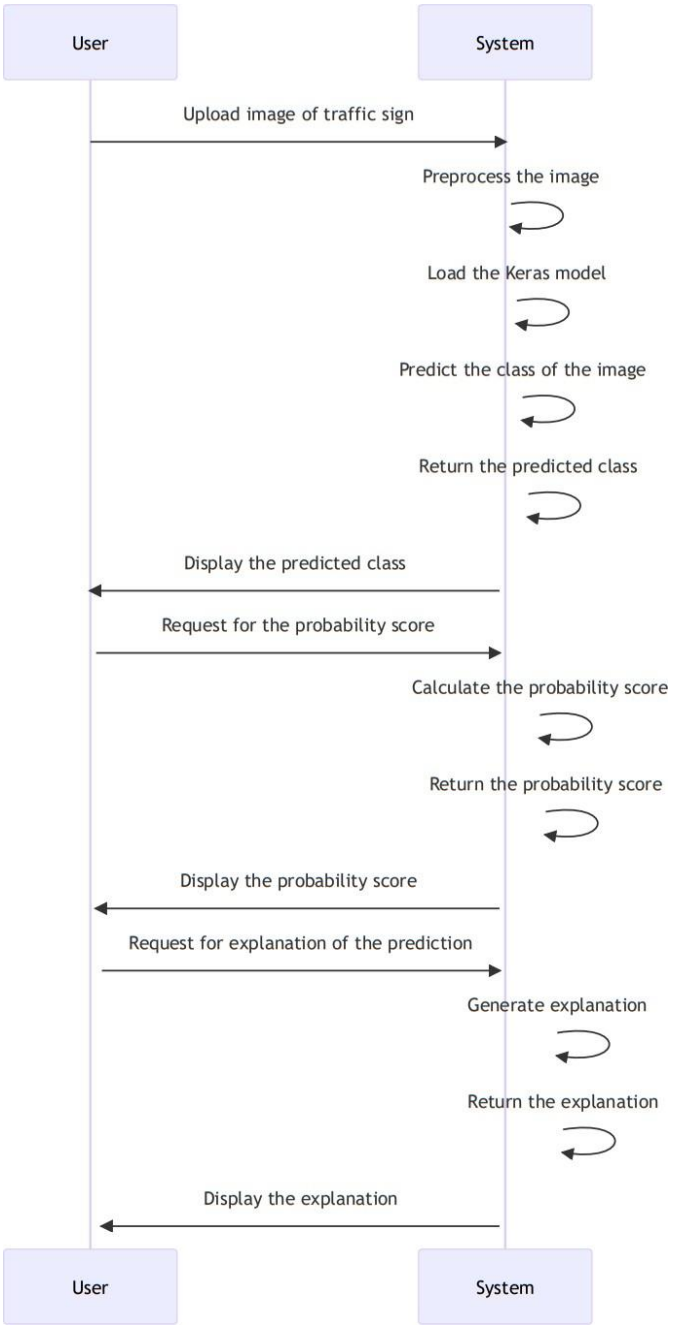

Fig. 6. User's interaction with the system

V. CONCLUSION AND FUTURE WORK

This paper presents a promising solution for enhancing road safety and reducing driver error-induced accidents through the use of a traffic sign recognition system developed using Convolutional Neural Networks (CNN). The system achieved high classification accuracy of 95% on a validation set and 94.5% on a real-world dataset, demonstrating its robustness to varying lighting and weather conditions and its ability to accurately recognize traffic signs at different distances and angles. The system comprises three main components, namely image pre-processing, CNN model, and post-processing, which have been described in detail. The integration of this system with autonomous vehicles can significantly improve their safety and reliability by enabling them to make informed decisions on the road. Additionally, the system's implementation has shown promising results in recognizing various traffic signs, and its performance can be further improved by expanding the dataset, optimizing its real-time performance, and testing it in real-world scenarios. The project has identified some essential considerations, such as detecting signs at night using non-infrared webcams and impaired detection due to reflections or impaired sign shapes. A text-to-speech module can be added to the model to reduce the occurrence of accidents during the day and night by making it easier for drivers to focus on driving. The use of the German Traffic Sign Recognition Benchmark (GTSRB) [15] in the project has made it more accurate and faster, and the minimal hardware used has reduced costs while being free from hardware impairments. Overall, the traffic sign recognition system developed using CNN technology holds immense potential to enhance road safety by accurately identifying traffic signs and alerting drivers and pedestrians in real-time. Its implementation can significantly contribute to increasing road safety and reducing the number of accidents on the roads.

There are several avenues for future research and development of the traffic sign recognition system using CNN technology. Firstly, efforts can be made to increase the accuracy of the model in all scenarios, including poor weather conditions or partially obscured signs. This can be achieved by exploring different techniques and algorithms that can enhance the model's performance. Expanding the dataset to include a larger variety of traffic signs from different countries can further improve the model's accuracy and reliability. This can also help to identify new categories of signs that may be important for road safety. Enhancing real-time performance is critical for practical applications. Therefore, optimizing the model to reduce inference time and improve its efficiency can improve its real-time performance. Testing the model in real-world scenarios is crucial for evaluating its effectiveness and identifying areas where improvements can be made. Real-world testing can also help determine the model's reliability and accuracy in different weather conditions and environments. Integrating the traffic sign recognition system with autonomous vehicles can further improve their safety and reliability. This can be achieved by providing the necessary inputs for the vehicle to recognize traffic signs and make appropriate decisions, such as stopping at a red light or yielding to pedestrians. Therefore, further research and development in this area can lead to a safer and more efficient transportation system.

REFERENCES

[1]. Sermanet, P., Kavukcuoglu, K., Chintala, S., & LeCun, Y. (2011). Traffic sign recognition with multi-scale convolutional networks. Proceedings of the International Joint Conference on Neural Networks (IJCNN).
[2]. Zhang, Y., Li, S., & Zhu, J. (2016). Traffic-sign recognition using a dynamic network reconfiguration module. IEEE Transactions on Intelligent Transportation Systems, 17(12), 3451-3462.

[3]. Li, J., Song, Q., & Zhang, X. (2019). A hybrid approach for traffic sign recognition based on convolutional neural network and hand-crafted features. Journal of Visual Communication and Image Representation, 58, 454-460.
[4]. Zhu, X., Liu, X., Liang, J., & Xu, M. (2018). A VGG-16 based transfer learning approach for traffic sign recognition. IEEE Access, 6, 25509-25517.
[5]. Mohan, R., Ramanathan, K., & Srinivasan, R. (2017). Real-time traffic sign recognition using transfer learning. Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC).
[6]. Liu, Y., Gao, Y., & Ji, R. (2018). Multi-scale convolutional neural network for traffic sign recognition. IEEE Transactions on Intelligent Transportation Systems, 19(11), 3470-3479.
[7]. Zhang, Y., Zhang, W., & Liu, M. (2018). Attention-based convolutional neural network for traffic sign recognition. IEEE Transactions on Intelligent Transportation Systems, 19(7), 2204-2213.
[8]. Zhang, J., Tang, Y., Xie, Y., & Yao, H. (2019). Domain adaptation for traffic sign recognition using adversarial training. Journal of Visual Communication and Image Representation, 59, 148-156.
[9]. Shustanov, A., & Yakimov, P. (2017). CNN design for real-time traffic sign recognition. Procedia engineering, 201, 718-725.
[10]. Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, August). Understanding of a convolutional neural network. In 2017 international conference on engineering and technology (ICET) (pp. 1-6). Ieee.
[11]. Ajit, A., Acharya, K., & Samanta, A. (2020, February). A review of convolutional neural networks. In 2020 international conference on emerging trends in information technology and engineering (ic-ETITE) (pp. 1-5). IEEE.
[12]. Abdel-Hamid, O., Mohamed, A. R., Jiang, H., & Penn, G. (2012). Convolutional neural networks for speech recognition. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 20(1), 33-42.
[13]. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
[14]. Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In Proceedings of the International Conference on Learning Representations (ICLR)
[15]. Gökberk, A. Y., DURDU, A., & NESİMİOĞLU, B. S. Accuracy Comparison of CNN Networks on GTSRB Dataset. Journal of Artificial Intelligence and Data Science, 2(2), 63-68.
[16]. Ziebinski, A., Cupek, R., Grzechca, D., & Chruszczyk, L. (2017, November). Review of advanced driver assistance systems (ADAS). In AIP Conference Proceedings (Vol. 1906, No. 1, p. 120002). AIP Publishing LLC.
[17]. Retallack, A. E., & Ostendorf, B. (2019). Current understanding of the effects of congestion on traffic accidents. International journal of environmental research and public health, 16(18), 3400.