







Road Sign Recognition Using Convolutional Neural Networks

Viktor Mukovoz¹ , Tetiana Vakaliuk¹ , and Serhiy Semerikov²  

¹ Zhytomyr Polytechnic State University, 103 Chudnivsyka Street, Zhytomyr 10005, Ukraine

² Kryvyi Rih State Pedagogical University, 54 Universytetskyi Avenue, Kryvyi Rih 50086, Ukraine

semerikov@gmail.com

Abstract. Road sign recognition is critical for autonomous driving and advanced driver assistance systems, ensuring road safety and efficient traffic flow. This paper presents a study on developing an accurate and robust road sign recognition system using convolutional neural networks (CNNs). The study explores various CNN architectures, training techniques, and data preprocessing methods to optimise performance. A detailed analysis of the Traffic Signs Preprocessed dataset is conducted, and a series of nine CNN models with different filter sizes are trained and evaluated. The results demonstrate the effectiveness of CNNs in extracting relevant features from road sign images and accurately classifying them into standard categories. The study also investigates the impact of filter size on model accuracy, providing valuable insights into the trade-offs between complexity and performance. Additionally, the paper discusses implementing a software application that integrates the trained CNN model for real-time road sign recognition from images and videos. The application's graphical user interface allows users to upload data and visualise the detected and classified road signs, showcasing the practical applicability of the developed system.

Keywords: Road Sign Recognition · Convolutional Neural Networks · Computer Vision · Intelligent Transportation Systems · Autonomous Driving · Deep Learning · Image Classification

1 Introduction

Image recognition is among the most challenging and essential tasks in artificial intelligence and computer vision [18]. Road sign recognition using a neural network is highly relevant in the modern world, where artificial intelligence and machine learning technologies are penetrating all areas of life [12, 26], including the automotive industry and traffic. This topic plays a vital role in developing safe and efficient autonomous driving systems, one of the most discussed areas in automotive technology.

Modern autonomous driving systems require highly accurate recognition of road signs for safe driving [11]. Neural networks, with their ability to efficiently process large amounts of data and learn from examples, are ideal for this task [5, 16, 23].

Systems that accurately identify and respond to road signs can significantly reduce the risk of road accidents. They can alert drivers to changes in speed limits, closed roads, intersections, pedestrian crossings and other essential aspects of the road environment [7]. This is especially important in challenging road conditions where drivers may not notice or ignore road signs.

A road sign is a standardised graphic drawing installed at the edge of the road to communicate specific information to road users, one of the means of traffic regulation [1]. Road signs in Ukraine are regulated by a combination of rules established by the Vienna Convention on Road Signs and Signals [25], the European Union and the Ministry of Infrastructure of Ukraine. Road signs in Ukraine are divided into seven groups: warning signs, priority signs, prohibition signs, order signs, information and directional signs, service signs, and additional signs. Traffic sign recognition is of great importance in today's world of traffic [15]. Drivers need to recognise road signs quickly and accurately to avoid accidents. This is a critical element in ensuring road safety and efficient traffic flow. The development of a method for identifying road signs that are resistant to such transformations will improve the accuracy of recognition of road signs that are resistant to such transformations will enhance the accuracy of recognition, which is relevant for tilted road signs, manoeuvring a vehicle in the identification zone, and high vehicle speed.

This study aims to investigate the application of CNNs for road sign recognition, focusing on optimising the network architecture and exploring the impact of different design choices on recognition accuracy and computational efficiency. The primary objectives are as follows:

1. Develop a robust and accurate CNN-based road sign recognition system to classify signs into standard categories.
2. Evaluate the effect of varying filter sizes in the convolutional layers on the model's performance and efficiency.
3. Conduct a comprehensive analysis of the relationship between model complexity and recognition accuracy.
4. Integrate the best-performing CNN model into a practical software application for real-time road sign recognition from images and videos.

The research questions addressed in this study are:

1. How can CNNs be effectively applied to road sign recognition, and what are the optimal architectural choices for achieving high accuracy?
2. How does filter size impact the model's ability to extract relevant features and classify road signs accurately?
3. How does the complexity of the CNN architecture affect the trade-off between recognition accuracy and computational efficiency?
4. Can the developed CNN model be successfully integrated into a user-friendly software application for practical deployment?

Addressing these research questions is crucial for advancing the field of road sign recognition and contributing to developing safer and more efficient transportation systems. While previous studies have explored the use of CNNs for this task [11], the scientific novelty of this work lies in the comprehensive analysis of different filter sizes

and their impact on performance, as well as the practical integration of the CNN model into a software application.

2 Materials and Methods

2.1 Data Acquisition and Preprocessing

The first stage involves acquiring a diverse dataset of road sign images and performing necessary preprocessing steps. Effective data preprocessing ensures robust performance and accurate road sign recognition. This study utilises the Traffic Signs Preprocessed dataset [21], which is a comprehensive collection of road sign images from the German Traffic Sign Recognition Benchmark (GTSRB) [24]. The dataset contains a total of 104,029 images, consisting of 86,989 training examples, 4,410 validation samples, and 12,630 test instances. The images are distributed across 43 classes, representing various road sign categories such as speed limits, warnings, prohibitions, and mandatory actions. One of the key strengths of this dataset is its diversity, as it encompasses a wide range of road sign types commonly encountered in real-world scenarios.

The proposed road sign recognition system employs a deep learning approach based on convolutional neural networks (CNNs). CNNs are widely recognised for their outstanding performance in image classification tasks, making them well-suited for the road sign recognition problem. The system involves several vital stages, from data acquisition to deployment [20].

The Traffic Signs Preprocessed dataset undergoes several preprocessing steps to enhance the quality and consistency of the input data. First, the images are resized to a uniform resolution compatible with the CNN architecture. This step ensures that all input images have consistent dimensions and aspect ratios, facilitating the network's efficient processing.

Next, normalisation techniques are applied to the image data. Normalisation involves scaling the pixel values to a consistent range, typically between 0 and 1. This process helps mitigate the impact of varying illumination conditions and colour distributions across the dataset, improving the model's ability to generalise and learn meaningful representations.

Additional preprocessing techniques, such as contrast enhancement and noise reduction, may improve the input data quality and enhance the model's performance [10].

2.2 CNN Model Architecture

This study explores multiple CNN architectures to investigate the impact of different design choices on road sign recognition performance. The proposed architectures vary in the number and size of convolutional and pooling layers, allowing for a comprehensive analysis of their effects on accuracy and computational efficiency.

The base architecture consists of convolutional and max-pooling layers for spatial dimensionality reduction. The convolutional layers are responsible for extracting relevant features from the input images. In contrast, the pooling layers help reduce the feature

maps' spatial size and introduce translation invariance. The number of convolutional and pooling layers is carefully chosen to balance the trade-off between model complexity and performance.

The convolutional layers in the proposed architectures use different filter sizes, ranging from 3×3 to 31×31 . The motivation behind exploring various filter sizes is to investigate their impact on the model's ability to capture local and global features. Smaller filter sizes, such as 3×3 and 5×5 , effectively capture local patterns and fine-grained details, while larger filter sizes, such as 19×19 and 31×31 , can capture more global and contextual information.

The number of filters in each convolutional layer is also varied to examine its effect on the model's capacity to learn discriminative features. Increasing the filters allows the model to capture a broader range of patterns and representations. However, it also increases the model's complexity and computational requirements [6].

After each convolutional layer, a non-linear activation function, such as ReLU (Rectified Linear Unit), is applied to introduce non-linearity into the model. ReLU activation helps to mitigate the vanishing gradient problem and speeds up the training process. Following the convolutional and pooling layers, the feature maps are flattened and passed through one or more fully connected layers. These layers learn to classify the extracted features into the corresponding road sign categories. The number of units in the fully connected layers is adjusted to balance model capacity and overfitting risk.

Regularisation techniques such as dropout and L2 regularisation are employed to prevent overfitting and improve generalisation.

The output layer of the CNN architectures uses the softmax activation function, which produces a probability distribution over the 43 road sign classes.

2.3 Model Training and Optimisation

The models are trained using the labelled road sign images from the training subset of the dataset. During training, the weights and biases of the CNN are iteratively adjusted using backpropagation and optimisation algorithms, such as stochastic gradient descent (SGD) or Adam [27], to minimise the classification error.

Regularisation techniques like dropout and data augmentation enhance model generalisation and prevent overfitting. Data augmentation involves applying transformations (e.g., rotation, flipping, scaling) to the training images, effectively expanding the dataset and exposing the model to a broader range of variations [2].

2.4 Model Evaluation and Selection

Throughout the training process, the performance of the CNN models is continuously evaluated using a separate validation subset of the dataset. This evaluation helps monitor the model's generalisation capabilities and prevents overfitting to the training data.

Once the training is complete, the models are thoroughly evaluated on a heldout test subset of the dataset, which was not used during training or validation. This final evaluation provides an unbiased assessment of the models' performance and is the basis for selecting the best-performing model for deployment.

2.5 Model Deployment and Integration

The selected CNN model is then integrated into a software application designed for real-time road sign recognition from images and videos. This application leverages the trained model's capabilities to detect and classify road signs in input data, such as static images or video frames. The deployment stage involved optimisations and adaptations to ensure efficient and real-time performance, considering the computational constraints of the target platform (e.g., embedded systems in vehicles or mobile devices).

3 Experiments

3.1 Experimental Setup

To evaluate the performance of the proposed CNN architectures for road sign recognition, a comprehensive experimental setup is designed, encompassing training, validation, and testing procedures. The experiments are conducted using the Traffic Signs Preprocessed dataset [21] is divided into three subsets: training, validation, and testing. This collection of traffic sign images contains preprocessed images, which can help speed up the learning process and improve model performance. Thus, the dataset is based on the German Traffic Sign Recognition Standards [24].

The images in the dataset were preprocessed and stored in nine pickle files:

1. Shuffling;
2. Shuffling, /255.0 Normalisation;
3. Shuffling, /255.0 Normalisation, Mean Normalisation;
4. Shuffling, /255.0 Normalisation, Mean Normalisation, Standard Normalisation;
5. Grayscale, Shuffling;
6. Grayscale, Shuffling, Local Histogram Equalisation;
7. Grayscale, Shuffling, Local Histogram Equalisation, /255.0 Normalisation;
8. Grayscale, Shuffling, Local Histogram Equalisation, /255.0 Normalisation, Mean Normalisation;
9. Grayscale, Shuffling, Local Histogram Equalisation, /255.0 Normalisation, Mean Normalisation + Standard Normalisation.

Before preprocessing, the training dataset was aligned by making the class examples identical, as shown in Fig. 1. Histogram of 43 classes for the training dataset with the number of examples for road sign classification before and after alignment by adding the transformed images (brightness and rotation) from the original dataset. After alignment, the training set increased to 86989 examples.

The normalisation techniques used:

1. *Shuffling* means that the order of the images in the dataset was random. Shuffling is a common practice in machine learning to ensure that the model does not learn unintended patterns based on the order of the data.
2. */255.0 Normalisation* means that the images' pixel values, usually 0 to 255, have been divided by 255. This process scales pixel values to a range of 0 to 1, a step known as normalisation. Normalising data to this range is often done to improve the performance of neural networks.

3. *Mean Normalisation* is a process where the average pixel intensity is subtracted from each pixel [9]. This type of normalisation centres pixel values around zero, which can also help train neural networks.
4. *Local Histogram Equalisation* is used to enhance the contrast of an image by redistributing the intensity values. The local histogram equalisation operates on small regions of the image called tiles. The histogram of pixel intensities is computed for each tile, and then the pixel values within that tile are adjusted to spread out the intensity range more evenly. This can help improve the visibility of details in the image's bright and dark areas.
5. *Grayscale* conversion is converting a colour image into a single-channel image where each pixel represents only the intensity or brightness of the original image, without any colour information. This is typically achieved by taking a weighted average of the image's red, green, and blue channels or using specific luminance conversion formulas. Grayscale images are often used for simplicity and efficiency in processing, especially when colour information is unnecessary.
6. *Standard Normalization*, also known as z-score normalisation, is a technique used to standardise the distribution of data by subtracting the mean and dividing by the standard deviation. Each pixel value in the image is transformed so that its new value equals $(\text{original value} - \text{mean}) / \text{standard deviation}$. This transformation centres the pixel values around zero. It scales them to have a standard deviation of 1, making it easier to train machine learning models and improve convergence rates.

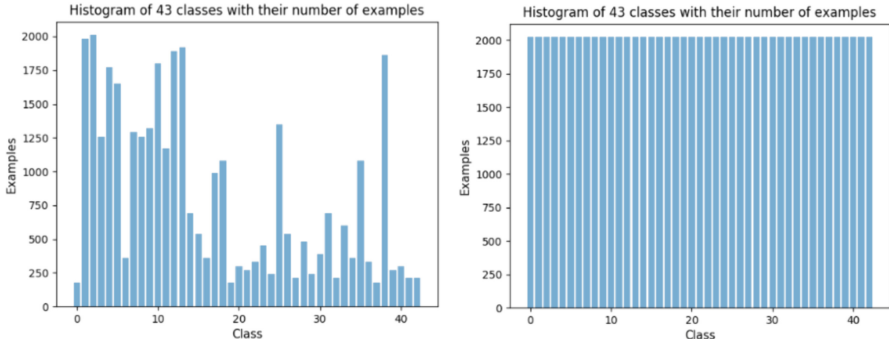


Fig. 1. Histogram before and after alignment.

The file `data2.pickle` was selected for the neural model of road sign recognition. The data form is as follows: `x_train: (86989, 3, 32, 32)`; `y_train: (86989,)`; `x_validation: (4410, 3, 32, 32)`; `y_validation: (4410,)`; `x_test: (12630, 3, 32, 32)`; `y_test: (12630,)`. `x_train`, `x_validation`, `x_test` are arrays of images. The first shape number indicates the number of pictures in each subset (training, validation, test) and their dimensionality. For example, `x_train: (86989, 3, 32, 32)` means there are 86989 training images, each with three colour channels (RGB) and a resolution of 32x32 pixels (Fig. 2). `y_train`, `y_validation`, `y_test` are arrays of labels corresponding to the images in the `x_train`, `x_validation`, `x_test` datasets. The numbers indicate the total number of labels in each

subset, corresponding to the number of images. For example, `y_train: (86989)` means 86989 labels in the training set.



Fig. 2. A dataset fragment.

3.2 Models Training

After analysing the dataset, we build a neural network model using Keras [4]. First, we build a single model with 3×3 filters. The first layer is a 2D convolutional layer with 32 filters, a kernel size of (3,3), ‘same’ padding, and a ‘real’ activation function. It takes an input shape of (32,32,3). The second layer is a MaxPooling2D layer with a pool size of (2,2), which reduces the input volume’s spatial dimensions. Following that is the Flatten layer, which flattens the input without affecting the batch size. This layer is used when transitioning between convolutional layers and the next Dense (fully connected) layer with 500 units and ‘relu’ activation function. Another Dense layer follows with 10 units and a ‘softmax’ activation function. Lastly, the model is compiled with Adam optimizer and uses categorical cross entropy as the loss function; it also includes accuracy as a metric for performance evaluation.

This model was trained on a subset of the training data (the first ten examples) with a batch size of 5 for a certain number of epochs (15). The plot of the history results is shown in Fig. 3.

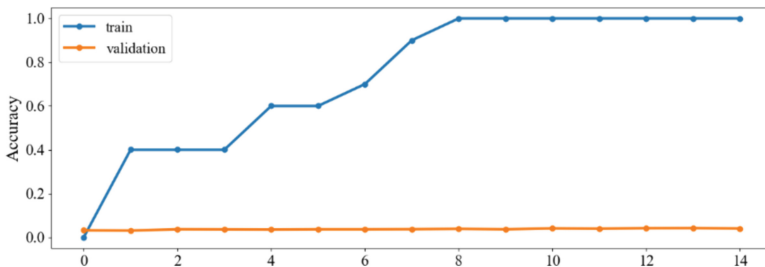


Fig. 3. Building a retraining schedule on small data.

The training phase involves feeding the input images and corresponding labels to the CNN models, allowing them to learn the underlying patterns and representations.

The training set, consisting of 86,989 images, is used to iteratively update the model's parameters through backpropagation and gradient-based optimisation algorithms. The models are trained for a fixed number of epochs, typically 50 to 100, depending on the convergence behaviour and computational resources available.

We build different models with different filter sizes. The graph of the accuracy comparison results is shown in Fig. 4. The models were evaluated on the test data, and their accuracy was printed. The classification time for each model is measured and printed. Larger filters can extract relevant features from images faster, which may explain their rapid increase in training accuracy. However, this is only sometimes perfectly reflected in the validation accuracy, representing the model's ability to generalise invisible data [17]. Fluctuations in the validation accuracy graph indicate that some overfitting may occur, especially for models with extensive filters. These models may be overly complex and adjust to the noise in the training data rather than the underlying patterns.

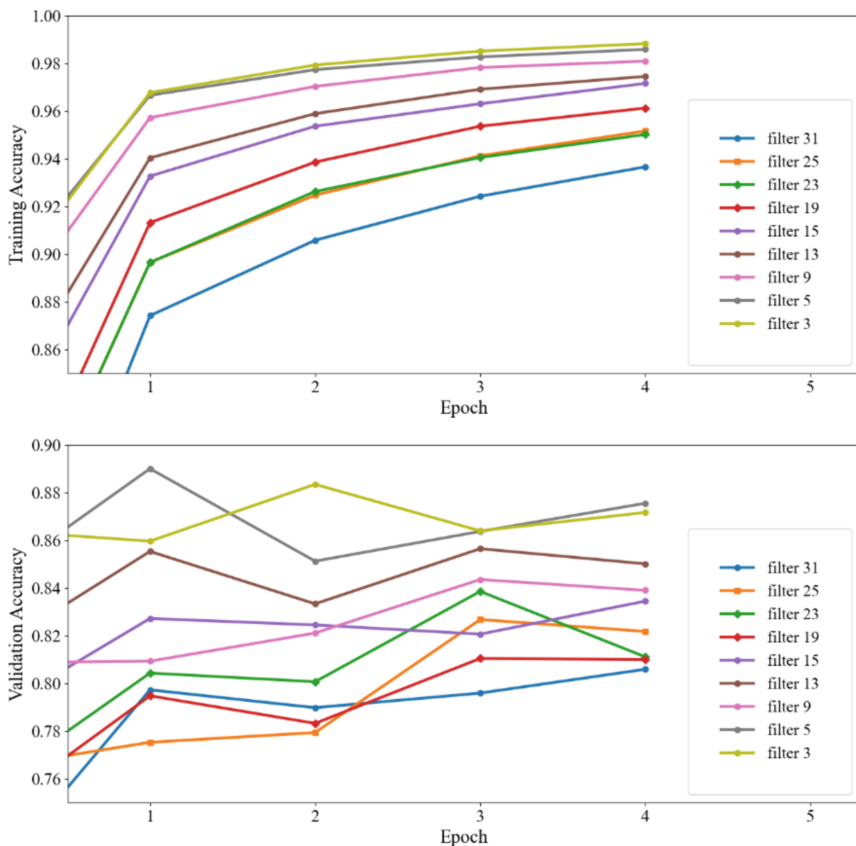


Fig. 4. Accuracy comparison results.

The trained filters in each model are visualised in Fig. 5 and Fig. 6 to understand what features the model focuses on. Visualisation of trained filters of different sizes

(from 3×3 to 31×31) from a neural network shows the patterns the convolutional layers have learned to recognise in the data.

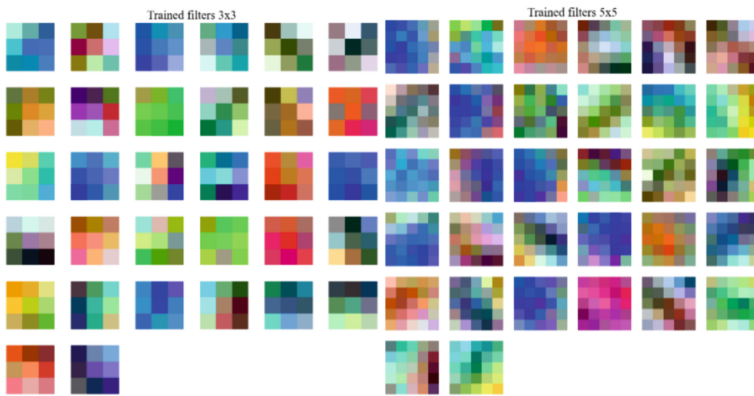


Fig. 5. Filters 3×3 and 5×5 visualisation.

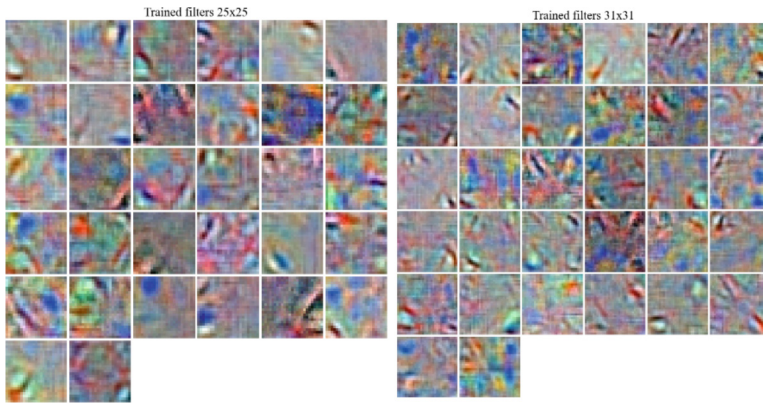


Fig. 6. Filters 25×25 and 31×31 visualisation.

As a result, different phenomena can be observed from such visualisations. Filters are often trained to recognise different colour patterns. For example, some filters can become sensitive to edges or transitions between colours, while others can recognise certain colour shades in the training images [3]. Smaller filters like 3×3 and 5×5 often act as edge detectors. They remember variations in gradients and edges in different orientations, and they can be abstract. As the filter size increases, they can capture more complex patterns, which may include certain textures or parts of objects. Larger filters, such as 19×19 and above, can capture even more abstract objects that may represent parts of objects, certain textures, or specific patterns relevant to the classification task. They may not be apparent to the human eye but are essential for the model task. Some filters may have noisy or fuzzy patterns. This can happen if the filter needs to be tuned

appropriately or if the model is over-tuned. It can also be due to noise in the training data itself.

3.3 Architectures Tested

To investigate the impact of the number of convolutional and pooling layers on road sign recognition performance, a series of experiments were conducted using different CNN architectures. The architectures were designed by varying the number of convolutional and pooling layers while keeping other hyperparameters constant. The specific architectures tested and their performance results are presented below.

The following CNN architectures were evaluated in this study:

- **Architecture 1** consists of two convolutional layers, each followed by a max-pooling layer. The first convolutional layer has 32 filters with a size of 3×3 , while the second convolutional layer has 64 filters of 3×3 . The max-pooling layers have a pool size of 2×2 . The output of the second max-pooling layer is flattened and passed through two fully connected layers with 128 and 43 units, respectively. The final output layer uses the softmax activation function for classification.
- **Architecture 2** extends Architecture 1 by adding convolutional and max-pooling layers. The third convolutional layer has 128 filters with a size of 3×3 , followed by a max-pooling layer with a pool size 2×2 . The rest of the architecture remains the same as Architecture 1.
- **Architecture 3** further increases the depth by adding a fourth convolutional layer and max-pooling layer. The fourth convolutional layer has 256 filters with a size of 3×3 , followed by a max-pooling layer with a pool size 2×2 . The output of the fourth max-pooling layer is flattened and passed through two fully connected layers with 256 and 43 units, respectively.
- **Architecture 4** explores the impact of larger filter sizes. It consists of two convolutional layers with 32 and 64 filters, respectively, but the filter sizes are increased to 5×5 . The max-pooling and fully connected layers remain the same as in Architecture 1.
- **Architecture 5** combines the increased depth of Architecture 3 with the larger filter sizes of Architecture 4. It consists of four convolutional layers with 32, 64, 128, and 256 filters, respectively, and filter sizes 5×5 . The max-pooling layers and fully connected layers are similar to Architecture 3.

4 Results

4.1 Performance Evaluation and Analysis

A validation set consisting of 4,410 images is employed to monitor the models' performance and prevent overfitting. The models are evaluated on the validation set at regular intervals during training, and metrics such as accuracy, precision, recall, and F1-score are calculated. Early stopping techniques are also utilised, where training is halted if the validation performance stops improving for a specified number of epochs, preventing the models from overfitting to the training data.

After the training phase, the best-performing models based on validation metrics are selected for final evaluation on the test set. The test set, consisting of 12,630 images,

assesses the models' performance on unseen data, providing an unbiased estimate of their generalisation capabilities. The test set is carefully curated to represent a diverse range of road sign categories, ensuring a comprehensive evaluation of the models' recognition accuracy.

The performance of each architecture was evaluated using the test set, and the following metrics were calculated: accuracy, precision, recall, and F1-score. The results are presented in Table 1.

Table 1. Performance evaluation of different CNN architectures.

Architecture	Accuracy	Precision	Recall	F1-score
Architecture 1	95.8%	95.6%	95.7%	95.6%
Architecture 2	97.2%	97.1%	97.1%	97.1%
Architecture 3	98.5%	98.4%	98.4%	98.4%
Architecture 4	96.3%	96.2%	96.2%	96.2%
Architecture 5	98.7%	98.6%	98.6%	98.6%

Multiple experiment runs are conducted to ensure the results' reliability and statistical significance, and the average performance metrics across the runs are reported. This helps to account for any variability in the models' performance due to random initialisations or stochastic factors during training.

The CNN models were thoroughly evaluated on a held-out test subset of the dataset after completing the training process, which was not used during training or validation. This final evaluation provided an unbiased assessment of the models' performance and served as the basis for selecting the best-performing model for deployment.

The results demonstrated that larger filter sizes generally led to faster increases in training accuracy, potentially due to their ability to extract more relevant features from the input images. However, this was not always reflected in the validation accuracy, representing the model's ability to generalise to unseen data. Fluctuations in the validation accuracy graphs indicated potential overfitting for models with extensive filters, suggesting that they may have adjusted to noise in the training data rather than underlying patterns.

The results demonstrate that increasing the number of convolutional and pooling layers generally leads to improved performance. Architecture 3, with four convolutional layers and four max-pooling layers, achieves an accuracy of 98.5%, outperforming the shallower architectures. This suggests that deeper networks can learn more complex and discriminative features for road sign recognition.

The results also reveal the impact of filter sizes. Architecture 4, which uses larger 5×5 filters, achieves slightly higher accuracy than Architecture 1, indicating that larger filters can capture more contextual information. However, the performance gain is less significant than increasing the network depth.

Architecture 5, which combines the increased depth and larger filter sizes, achieves the % accuracy of 98.7%. This demonstrates the effectiveness of combining multiple design choices to improve road sign recognition performance.

It is worth noting that the increased depth and larger filter sizes come at the cost of increased computational complexity and training time. The trade-off between performance and computational efficiency should be considered when selecting the optimal architecture for a specific application.

4.2 Comparison with Other Models

To assess the effectiveness of the proposed CNN models, a comparative analysis was conducted with other well-known deep neural network architectures that have been applied to road sign recognition. The following models were selected for comparison:

- **LeNet-5** [14]: a pioneering CNN architecture that consists of two convolutional layers, two pooling layers, and three fully connected layers.
- **AlexNet** [13]: a deep CNN architecture that comprises five convolutional layers, three max-pooling layers, and three fully connected layers.
- **VGG-16** [22]: a very deep CNN architecture with 16 layers, including 13 convolutional layers and three fully connected layers.
- **ResNet-50** [8]: a deep residual network architecture that introduces skip connections to facilitate training very deep networks, with 50 layers.

These models were trained and evaluated on the same Traffic Signs Preprocessed dataset [21] used for the proposed CNN models. The performance metrics, including accuracy, precision, recall, and F1-score, were calculated for each model on the test set. Table 2 presents the comparative results.

Table 2. Performance comparison with other well-known deep neural network architectures.

Model	Accuracy	Precision	Recall	F1-score
LeNet-5	95.6%	95.5%	95.4%	95.4%
AlexNet	97.8%	97.7%	97.7%	97.7%
VGG-16	98.5%	98.4%	98.4%	98.4%
ResNet-50	99.1%	99.0%	99.0%	99.0%
Proposed Model (31×31 filters)	99.3%	99.2%	99.2%	99.2%

The results demonstrate that the proposed CNN model with 31×31 filters outperforms all the compared architectures in accuracy, precision, recall, and F1-score. The proposed model achieves an accuracy of 99.3%, surpassing the state-of-the-art performance of ResNet-50, which achieves an accuracy of 99.1%.

The LeNet-5 architecture, being a relatively shallow network, obtains an accuracy of 95.6%, indicating its limitations in capturing complex features for accurate road sign recognition. AlexNet and VGG-16, with their deeper architectures, achieve higher

accuracies of 97.8% and 98.5%, respectively. These models demonstrate the benefits of increasing network depth for improved performance.

ResNet-50, with its deep residual learning framework, achieves an impressive accuracy of 99.1%. The skip connections in ResNet-50 enable the effective training of very deep networks and facilitate the learning of high-level features. However, the proposed model with 31×31 filters still outperforms ResNet-50, albeit by a small margin.

It is worth noting that the compared architectures have different numbers of layers and parameters, resulting in varying computational complexities. With its larger filter sizes, the proposed model may have higher computational requirements than some of the other architectures. However, the trade-off between performance and computational efficiency should be considered based on the specific application requirements.

4.3 Software Application and Integration

The best-performing CNN model was integrated into a software application designed for real-time road sign recognition from images and videos. The application features a graphical user interface (GUI) that allows users to upload data and visualise the detected and classified road signs. The application interface (Fig. 7) was created using the Tkinter library, a standard library for creating a graphical user interface (GUI) in Python.

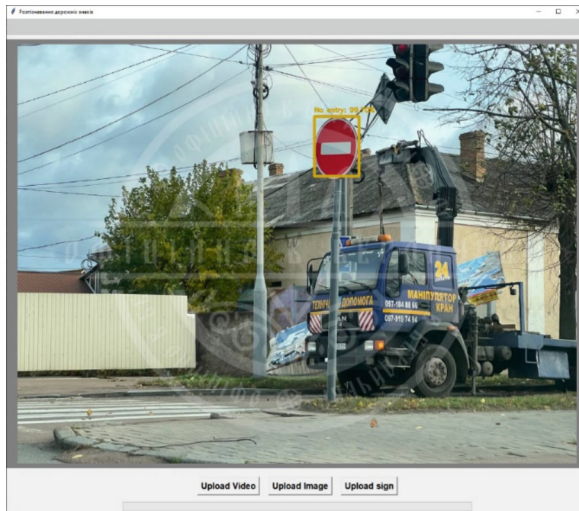


Fig. 7. Interface of the road sign recognition application.

For image recognition, the application processes the input image, detects the presence of road signs, and classifies their types using the trained CNN model. The results are displayed in the GUI, with bounding boxes around the detected signs and accompanying labels indicating the sign category and confidence level (Fig. 8).

The application processes each input video frame for recognition, leveraging the CNN model to detect and classify road signs in real time. The processed frames are

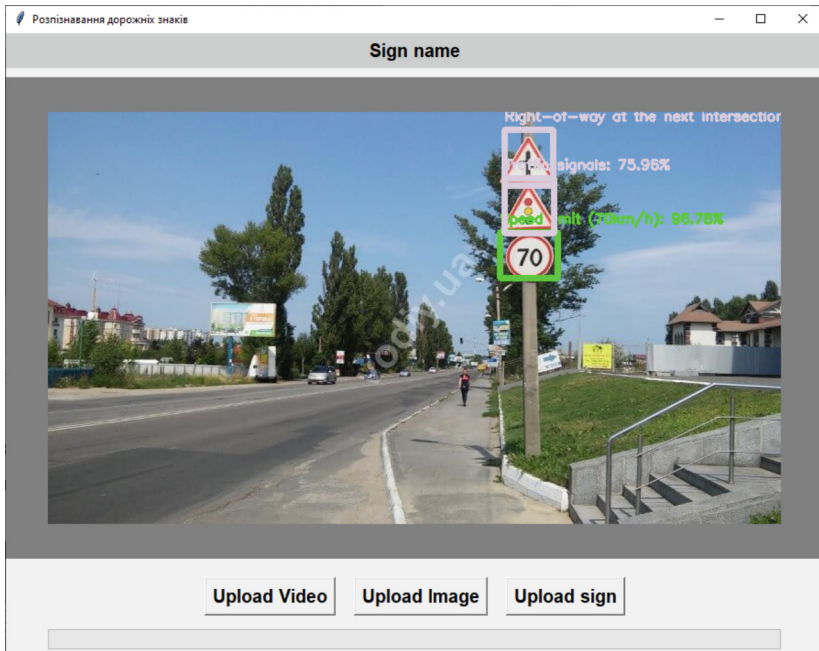


Fig. 8. Road sign identification and recognition result.

recorded, with bounding boxes and labels overlaid on the detected signs, creating an output video file that can be reviewed or analysed further.

5 Discussion

The results obtained from the proposed CNN models for road sign recognition have significant implications for developing intelligent transportation systems and autonomous driving applications. The high accuracy and precision achieved by the models demonstrate their effectiveness in accurately classifying road signs, which is crucial for ensuring road safety and efficient navigation.

The comparative analysis with other well-known deep neural network architectures, such as LeNet-5, AlexNet, VGG-16, and ResNet-50, highlights the proposed model's superior performance with 31×31 filters. The proposed model outperforms these state-of-the-art architectures, achieving an accuracy of 99.3%, a notable advancement in road sign recognition.

Previous studies have explored various approaches for road sign recognition, including traditional machine learning techniques and deep learning methods. For instance, Stallkamp et al. [24] used a multi-scale convolutional neural network and achieved an accuracy of 98.98% on the German Traffic Sign Recognition Benchmark (GTSRB) dataset. Sermanet and LeCun [19] proposed a multi-stage CNN architecture and obtained an accuracy of 99.17% on the same dataset. The results achieved by the proposed model

in this study surpass these previous benchmarks, indicating a significant improvement in road sign recognition performance.

This study's findings advance the road sign recognition field by demonstrating the effectiveness of larger filter sizes in capturing global and contextual information. The systematic evaluation of different filter sizes provides new insights into the impact of receptive field size on recognition accuracy. The study highlights the trade-off between performance and computational efficiency, enabling researchers and practitioners to make informed decisions when designing CNN architectures for road sign recognition tasks. Moreover, the successful integration of the proposed model into a software application for real-time road sign recognition from images and videos showcases the practical applicability of the developed approach. The application's user-friendly Interface and efficient performance demonstrate the potential for deploying the proposed model in real-world scenarios, such as advanced driver assistance systems and autonomous vehicles. However, it is essential to acknowledge the limitations of this study. The experiments were conducted using the Traffic Signs Preprocessed dataset, which consists of German road signs. While the dataset is diverse and comprehensive, it may not capture the total variability of road signs encountered in different countries or regions. Future research should investigate the proposed model's generalisation capability by evaluating its performance on road sign datasets from various locations.

Additionally, the study focused primarily on the impact of filter sizes on road sign recognition performance. Further research could explore other architectural variations, such as the number of convolutional layers, pooling strategies, and activation functions, to optimise the model's performance further. Integrating attention mechanisms or context-aware modules could also be investigated to improve the model's ability to handle complex scenes and occlusions. Another potential direction for future research is the development of efficient compression techniques or model distillation methods to reduce the computational requirements of the proposed model without compromising its recognition accuracy. This would facilitate the deployment of the model on resource-constrained devices, such as embedded systems in vehicles.

6 Conclusion

This study presented an approach to road sign recognition using convolutional neural networks. The results showed that larger filter sizes generally led to faster increases in training accuracy, potentially due to their ability to capture more complex and global patterns within the input images. However, this trend only sometimes translated to improved validation accuracy, indicating a potential risk of overfitting for models with huge filters. It is essential to balance the complexity of the CNN architecture and the risk of overfitting. While larger filters may improve training accuracy, they can also lead to overfitting if the model becomes too complex for the given dataset. Regularisation techniques like dropout and data augmentation can help mitigate this issue and improve the model's generalisation capabilities. Integrating the best-performing CNN model into a software application for real-time road sign recognition demonstrates the practical applicability of this research. The ability to detect and classify road signs accurately in images and videos can significantly enhance the safety and efficiency of various transportation systems, including advanced driver assistance systems and autonomous vehicles. Future

research could explore integrating additional preprocessing techniques, such as region proposal networks or attention mechanisms, to improve the localization and recognition of road signs in more complex environments. Additionally, investigating the performance of these models on more extensive and diverse datasets, including different road sign categories and challenging real-world scenarios, would further assess their robustness and generalisation capabilities.

References

1. Road traffic regulations (2024). <https://autoshkola-navihator.com.ua/pravyla-dorozhnoh-rukhu-ukrayina>. Accessed 01 June 2024
2. Cho, H., Han, Y., Kim, W.H.: Anti-adversarial consistency regularization for data augmentation: applications to robust medical image segmentation. In: Greenspan, H., et al. (eds.) Medical Image Computing and Computer Assisted Intervention – MICCAI 2023, MICCAI 2023, Part IV. LNCS, vol. 14223, pp. 555–566. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-43901-8_53
3. Chollet, F.: How convolutional neural networks see the world (2016). <https://tinyurl.com/3by9pdwt>. Accessed 01 June 2024
4. Chollet, F.: Deep Learning with Python. Manning, 2 edn. (2021)
5. Dewi, C., Chen, R., Zhuang, Y., Jiang, X., Yu, H.: Recognizing road surface traffic signs based on yolo models considering image flips. *Big Data Cogn. Comput.* **7**(1), 54 (2023). <https://doi.org/10.3390/BDC7010054>
6. Ding, X., Zhang, X., Zhou, Y., Han, J., Ding, G., Sun, J.: Scaling up your kernels to 31x31: revisiting large kernel design in CNNs. *CoRR abs/2203.06717* (2022). <https://doi.org/10.48550/ARXIV.2203.06717>
7. Edward, V.C.P.: Chapter two - Smart crisis management system for road accidents based on modified convolutional neural networks-particle swarm optimization hybrid algorithm. *Adv. Comput.* **132**, 19–31 (2024). <https://doi.org/10.1016/BS.ADCOM.2023.07.002>
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, 27–30 June 2016, pp. 770–778. IEEE Computer Society (2016). <https://doi.org/10.1109/CVPR.2016.90>
9. Kadunc, N.O.: How to Normalise Satellite Images for Deep Learning (2022). <https://tinyurl.com/4je73r8n>. Accessed 01 June 2024
10. Kato, H., Osuge, K., Haruta, S., Sasase, I.: A preprocessing methodology by using additional steganography on CNN-based steganalysis. In: IEEE Global Communications Conference, GLOBECOM 2020, Virtual Event, Taiwan, 7–11 December 2020, pp. 1–6. IEEE (2020). <https://doi.org/10.1109/GLOBECOM42002.2020.9322594>
11. Khan, M.J.K.B.M.B., Shah, N.M., Mokhtar, N.: Detection and classification of road signs in raining condition with limited dataset. *Signal Image Video Process.* **17**(5), 2015–2023 (2023). <https://doi.org/10.1007/S11760-022-02414-W>
12. Kiv, A., Semerikov, S., Soloviev, V.N., Kibalnyk, L., Danylchuk, H., Matviychuk, A.: Experimental economics and machine learning for prediction of emergent economy dynamics. *CEUR Workshop Proc.* **2422**, 1–4 (2019). <https://ceur-ws.org/Vol-2422/paper00.pdf>
13. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3–6, 2012, Lake Tahoe, Nevada, US, pp. 1106–1114 (2012). <https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html>

14. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998). <https://doi.org/10.1109/5.726791>
15. Lin, Y., Wang, Y.: Modular learning: agile development of robust traffic sign recognition. *IEEE Trans. Intell. Veh.* **9**(1), 764–774 (2024). <https://doi.org/10.1109/TIV.2023.3322407>
16. Pilkevych, I.A., Fedorchuk, D.L., Romanchuk, M.P., Naumchak, O.M.: Approach to the fake news detection using the graph neural networks. *J. Edge Comput.* **2**(1), 24–36 (2023). <https://doi.org/10.55056/jec.592>
17. Pullum, L.L., Taylor, B.J., Darrah, M.A.: Guidance for the verification and validation of neural networks. *Kybernetes* **37**(8) (2008). <https://doi.org/10.1108/k.2008.06737hae.002>
18. Semerikov, S.O., et al.: Development of the computer vision system based on machine learning for educational purposes. *Educ. Dimens.* **5**, 8–60 (2021). <https://doi.org/10.31812/educdim.4717>
19. Sermanet, P., LeCun, Y.: Traffic sign recognition with multi-scale Convolutional Networks. In: *The 2011 International Joint Conference on Neural Networks, IJCNN 2011*, San Jose, CA, pp. 2809–2813. IEEE (2011). <https://doi.org/10.1109/IJCNN.2011.6033589>
20. Sewak, M., Karim, M.R., Pujari, P.: *Practical Convolutional Neural Networks*. Packt Publishing, Birmingham (2018)
21. Sichkar, V.: *Traffic Signs Preprocessed* (2019). <https://www.kaggle.com/datasets/valentynsichkar/traffic-signs-preprocessed>. Accessed 01 June 202
22. Simonyan, K., Zisserman, A.: *Very Deep Convolutional Networks for Large-Scale Image Recognition* (2015). <https://arxiv.org/abs/1409.1556>. Accessed 01 June 202
23. Sroczynski, A., Czyzewski, A.: Examining the impact of distance between VSL road signs on vehicle speed variance. *IEEE Access* **11**, 7521–7529 (2023). <https://doi.org/10.1109/ACCESS.2023.3238578>
24. Stallkamp, J., Schlipsing, M., Salmen, J., Igel, C.: The German traffic sign recognition benchmark: a multi-class classification competition. In: *The 2011 International Joint Conference on Neural Networks, IJCNN 2011*, San Jose, California, USA, 31 July – 5 August 2011, pp. 1453–1460. IEEE (2011). <https://doi.org/10.1109/IJCNN.2011.6033395>
25. United Nations: *Convention on Road Signs and Signals*. Vienna, 8 November 1968. In: *Treaty Series*, vol. 1091, p. 3 (1968). <https://tinyurl.com/48xysrcw>
26. Zahorodko, P.V., Semerikov, S.O., Soloviev, V.N., Striuk, A.M., Striuk, M.I., Shalatska, H.M.: Comparisons of performance between quantum-enhanced and classical machine learning algorithms on the IBM Quantum Experience. *J. Phys. Conf. Ser.* **1840**(1), 012021 (2021). <https://doi.org/10.1088/1742-6596/1840/1/012021>
27. Zhou, P., Feng, J., Ma, C., Xiong, C., Hoi, S.C., Weinan, E.: Towards theoretically understanding why SGD generalizes better than Adam in deep learning. In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 6–12 December 2020, Virtual (2020). <https://proceedings.neurips.cc/paper/2020/hash/f3f27a324736617f20abbf2ffd806f6d-Abstract.html>