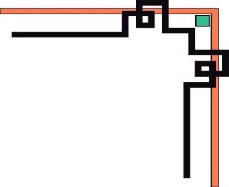
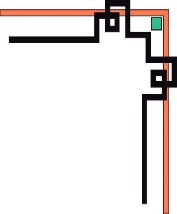
**HCMC UNIVERSITY OF TECHNOLOGY AND EDUCATION**



**FACULTY OF MECHANICAL ENGINEERING**

**'Y&&Y'**



**COURSE: Artificial Intelligence**

**Final Report**

**TOPIC:**

**CNN Design For Real-Time Traffic Sign Recognition**

**LECTURER: NGUYỄN TRƯỜNG THỊNH**

1. **LHP Code: 221ARIN331029E**

**STUDENTS:**

**NGUYỄN CÔNG QUÝ - 20134022**

**HỒ CHÍ MINH CITY , June 22, 2022**

**Contents**

[**1. Introduction about real-Time traffic sign recognition** 3](#_Toc124570875)

[**2. What is a convolutional neural network?** 5](#_Toc124570876)

[**3. Build and design a convolutional neural network** 12](#_Toc124570877)

[**4. Traffic Sign Localization and Tracking** 14](#_Toc124570878)

[**5. Training and Experimental Results** 15](#_Toc124570879)

[**6. Conclusions** 17](#_Toc124570880)

# **1. Introduction about real-Time traffic sign recognition**

The traffic signs are one of the most important elements in transportation infrastructure. They control the flow of traffic, give the right of way, inform a road user about the directions and distances along with a guide to a destination, or warn about dangerous places. In order to increase the safety and comfort of a driver, traffic sign recognition (TSR) systems have included advanced driver assistance systems as the crucial parts. Nowadays, commercial TSR systems are being deployed to cars for the purpose of recognizing speed limit signs or no-overtaking signs. So, these systems can easily warn the driver in certain situations, e.g. if the speed limit is being exceeded or if the traffic lane is being crossed at the place where the car must not cross over any continuous (unbroken) center line. Moreover, the TSR can be used for collecting traffic signs and locating GPS positions. Subsequently, the collected data are stored (incorporated into) to map databases of navigation systems or databases for local authorities [1]. Afterward, the navigation system can find the optimal route with respect to the restrictions given by the traffic signs such as speed, weight, height limits and etc. These databases of the traffic signs might be used as well by the local authorities to register all traffic signs in a region. The other way to process recognized traffic signs is to store them in the database distributed and replicated throughout the vehicular ad-hoc network. The information would therefore be accessible by any other road user and could be used to enhance road safety [2-4]. Currently, many publications could be found which propose various methods and techniques how to detect and recognise traffic signs. Classical methods of detection are based on colour segmentation in various colour spaces such as RGB, HSV, HSI, CIELab, CIELUV, CIECAM97s and others [5-8]. Some authors use other approaches based on the shape which are more robust with respect of different light conditions. There are no such problems with colours in pictures in case of bad illumination. Other possible approaches use machine learning algorithms, either for detection, classification or for both. Several types of algorithms are possible to use, e.g. artificial neural networks (ANN) [9], support vector machine (SVM) [5][10], boosting [11], genetic algorithms [12] and others. One exceptional survey of the state-of-the-art approaches for detection and recognition in this field you can find in work [13] was published by Meng-Yin Fu and Yuan-Shui Huang. In this paper, we will briefly introduce our proposed TSR system, our method of approximate position determining and a developed demonstrative application of collecting traffic signs to a database. In this work, we are mainly focused on the traffic signs approved by Vienna Convention [14] located in the Slovak Republic.

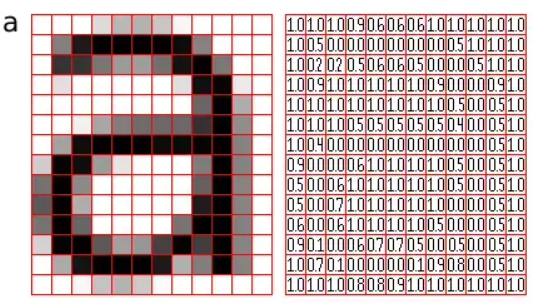


*Fig. 1. Images from GTSDB and GTSRB*

Moreover, the traffic sign recognition system is an important part of intelligent transportation systems, which can take images by the cameras installed on the front side of the motor vehicle and transfer the images to the image processing module of the system. It plays an important role in driver and pedestrian safety. The natural scene of traffic sign recognition often exists in complex scenes: disturbance, climate interference, illumination changes, dirty or blocked traffic sign, and skew distortions due to anthropogenic factors or disrepair. Therefore, the recognition of traffic signs is of great significance in natural scenes.

# **2. What is a convolutional neural network?**

Convolutional neural networks, also referred to as CNNs or ConvNets, are a subclass of neural networks that are particularly adept at processing data with a grid-like topology, like images. Binary visual data is represented as a digital image. It has a number of pixels that are arranged like a grid and are each assigned a value to indicate how bright and what color each pixel should be.

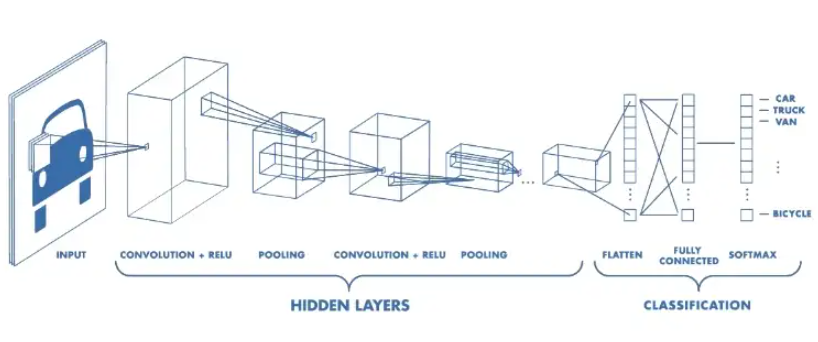


*Fig. 2. Representation of image as a grid of pixels*

The moment we see an image, the human brain begins processing a massive amount of data. Every neuron has a distinct receptive field and is connected to other neurons so that they collectively cover the entire visual field. Each neuron in a CNN processes data only in its receptive field, similar to how each neuron in the biological vision system responds to stimuli only in the constrained area of the visual field known as the receptive field. Lines, curves, and other simpler patterns are detected first by the layers, followed by more intricate patterns like faces and objects. One can enable sight to computers by using a CNN.

**Convolutional Neural Network Architecture**

A CNN typically has three layers: a convolutional layer, a pooling layer, and a fully connected layer.

****

*Fig. 3. Architecture of a CNN*

**Convolution Layer**

The convolution layer is the core building block of the CNN. It carries the main portion of the network’s computational load.

This layer performs a dot product between two matrices, where one matrix is the set of learnable parameters otherwise known as a kernel, and the other matrix is the restricted portion of the receptive field. The kernel is spatially smaller than an image but is more in-depth. This means that, if the image is composed of three (RGB) channels, the kernel height and width will be spatially small, but the depth extends up to all three channels.

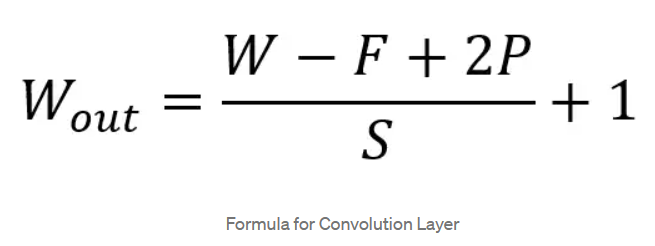
**eDiagram

Description automatically generated with medium confidence**

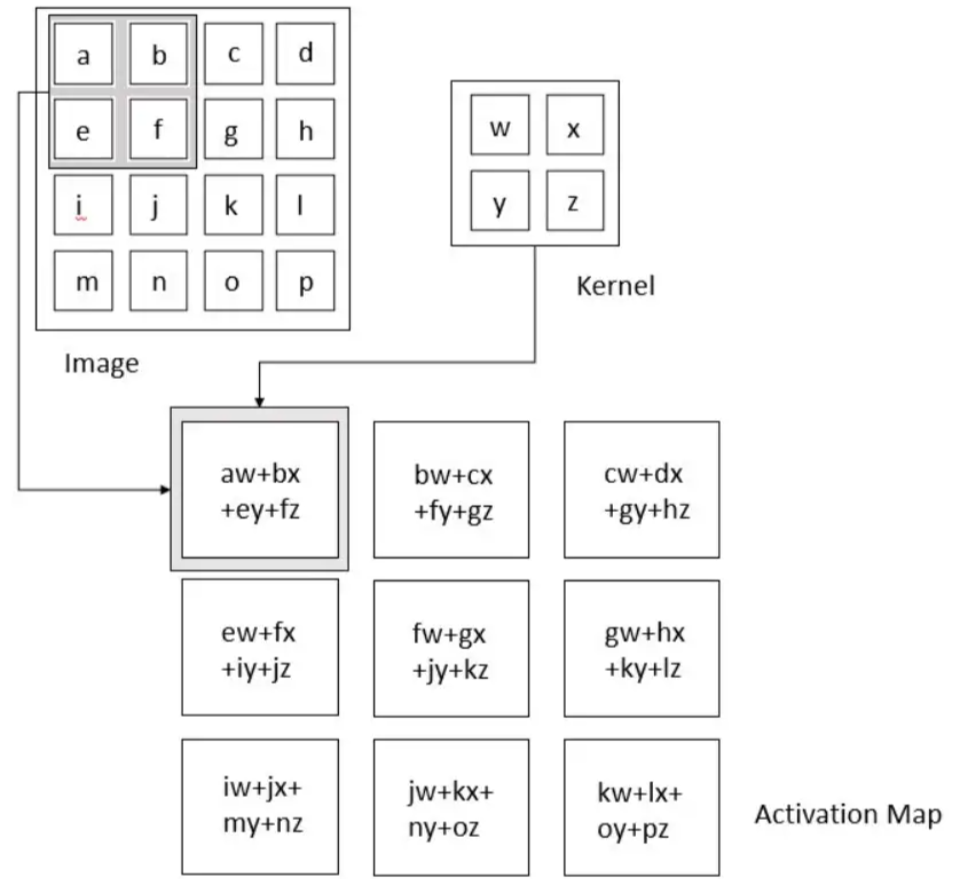
*Fig. 4. Illustration of Convolutional Operation*

During the forward pass, the kernel slides across the height and width of the image-producing the image representation of that receptive region. This produces a two-dimensional representation of the image known as an activation map that gives the response of the kernel at each spatial position of the image. The sliding size of the kernel is called a stride.

If we have an input of size W x W x D and Dout number of kernels with a spatial size of F with stride S and amount of padding P, then the size of output volume can be determined by the following formula:



This will yield an output volume of size Wout x Wout x Dout.



*Fig. 5. Convolution Operation*

**Motivation behind Convolution**

Convolution leverages three important ideas that motivated computer vision researchers: sparse interaction, parameter sharing, and equivariant representation. Let’s describe each one of them in detail.

Trivial neural network layers use matrix multiplication by a matrix of parameters describing the interaction between the input and output unit. This means that every output unit interacts with every input unit. However, convolution neural networks have sparse interaction. This is achieved by making kernel smaller than the input e.g., an image can have millions or thousands of pixels, but while processing it using kernel we can detect meaningful information that is of tens or hundreds of pixels. This means that we need to store fewer parameters that not only reduce the memory requirement of the model but also improve the statistical efficiency of the model.

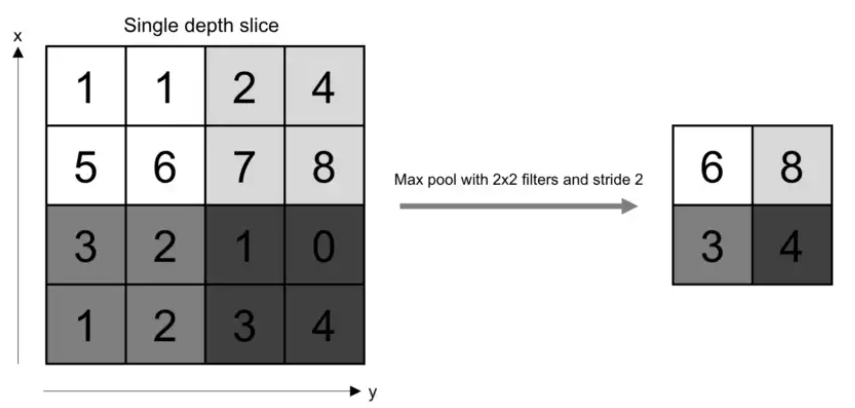
If computing one feature at a spatial point (x1, y1) is useful then it should also be useful at some other spatial point say (x2, y2). It means that for a single two-dimensional slice i.e., for creating one activation map, neurons are constrained to use the same set of weights. In a traditional neural network, each element of the weight matrix is used once and then never revisited, while the convolution network has shared parameters i.e., for getting output, weights applied to one input are the same as the weight applied elsewhere.

Due to parameter sharing, the layers of the convolution neural network will have a property of equivariance to translation. It says that if we changed the input in a way, the output will also get changed in the same way.

**Pooling Layer**

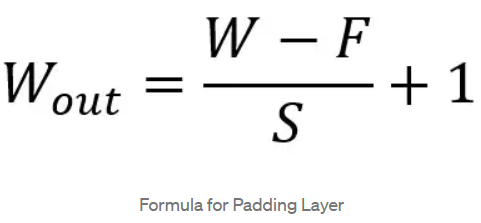
The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually.

There are several pooling functions such as the average of the rectangular neighborhood, L2 norm of the rectangular neighborhood, and a weighted average based on the distance from the central pixel. However, the most popular process is max pooling, which reports the maximum output from the neighborhood.



*Fig. 6. Pooling Operation*

If we have an activation map of size W x W x D, a pooling kernel of spatial size F, and stride S, then the size of output volume can be determined by the following formula:

**

This will yield an output volume of size Wout x Wout x D.

In all cases, pooling provides some translation invariance which means that an object would be recognizable regardless of where it appears on the frame.

**Fully Connected Layer**

Neurons in this layer have full connectivity with all neurons in the preceding and succeeding layer as seen in regular FCNN. This is why it can be computed as usual by a matrix multiplication followed by a bias effect.

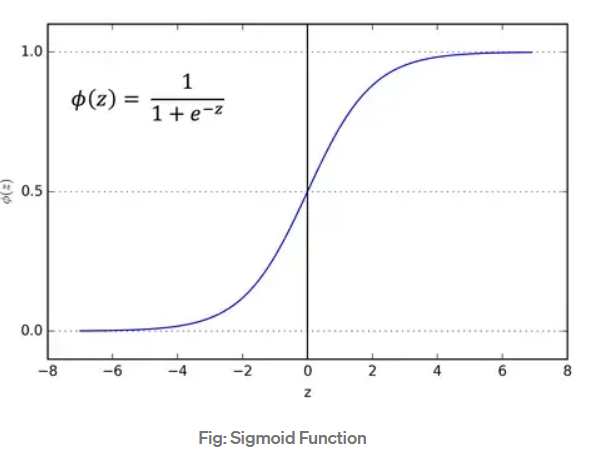
The FC layer helps to map the representation between the input and the output.

**Layers of Non-Linearity**

Non-linearity layers are frequently added right after the convolutional layer to add non-linearity to the activation map because convolution is a linear operation and images are anything but linear. Non-linear operations come in a variety of forms, the most common ones being:

**1. Sigmoid**

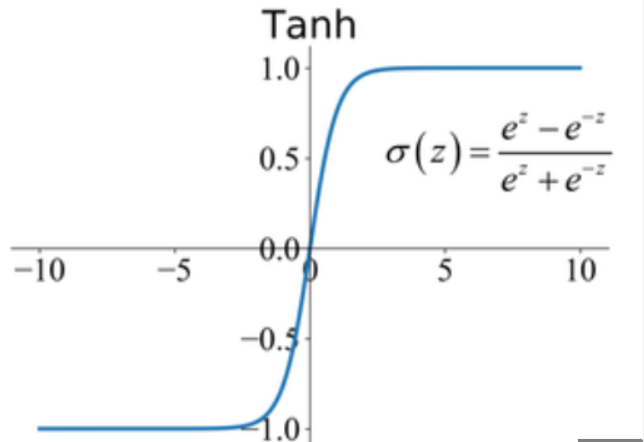
The mathematical formula for sigmoid nonlinearity is



It "squashes" a real-valued number into the range between 0 and 1. The gradient of a sigmoid becomes almost zero when the activation is at either tail, which is a very undesirable sigmoid property. Backpropagation will effectively "kill" the gradient if the local gradient gets too small.

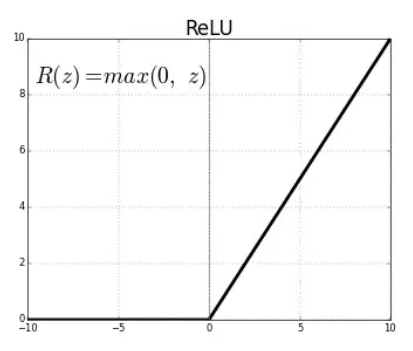
**2. Tanh**

Tanh squashes a real-valued number to the range [-1, 1]. Like sigmoid, the activation saturates, but — unlike the sigmoid neurons — its output is zero centered.

****

**3. RELU**

In recent years, the Rectified Linear Unit (ReLU) has gained a lot of popularity. It performs the function ƒ(κ)=max (0,κ) computation. Alternatively put, the activation is just a threshold at zero.



ReLU speeds up convergence by six times and is more dependable than sigmoid and tanh.

Unfortunately, be brittle during training, which is a drawback. It can be updated by a strong gradient that prevents the neuron from ever updating further. However, by choosing an appropriate learning rate, we can make this work.

# **3. Build and design a convolutional neural network**

def myModel():

    no\_Of\_Filters=60

    size\_of\_Filter=(5,5) # THIS IS THE KERNEL THAT MOVE AROUND THE IMAGE TO GET THE FEATURES.

                         # THIS WOULD REMOVE 2 PIXELS FROM EACH BORDER WHEN USING 32 32 IMAGE

    size\_of\_Filter2=(3,3)

    size\_of\_pool=(2,2)  # SCALE DOWN ALL FEATURE MAP TO GERNALIZE MORE, TO REDUCE OVERFITTING

    no\_Of\_Nodes = 512   # NO. OF NODES IN HIDDEN LAYERS

    model= Sequential()

    model.add((Conv2D(no\_Of\_Filters,size\_of\_Filter,input\_shape=(imageDimesions[0],imageDimesions[1],1),activation='relu')))  # ADDING MORE CONVOLUTION LAYERS = LESS FEATURES BUT CAN CAUSE ACCURACY TO INCREASE

    model.add((Conv2D(no\_Of\_Filters, size\_of\_Filter, activation='relu')))

    model.add(MaxPooling2D(pool\_size=size\_of\_pool)) # DOES NOT EFFECT THE DEPTH/NO OF FILTERS

    model.add((Conv2D(no\_Of\_Filters//2, size\_of\_Filter2,activation='relu')))

    model.add((Conv2D(no\_Of\_Filters // 2, size\_of\_Filter2, activation='relu')))

    model.add(MaxPooling2D(pool\_size=size\_of\_pool))

    model.add(Dropout(0.5))

    model.add(Flatten())

    model.add(Dense(no\_Of\_Nodes,activation='relu'))

    model.add(Dropout(0.5)) # INPUTS NODES TO DROP WITH EACH UPDATE 1 ALL 0 NONE

    model.add(Dense(noOfClasses,activation='softmax')) # OUTPUT LAYER

    # COMPILE MODEL

    model.compile(Adam(lr=0.001),loss='categorical\_crossentropy',metrics=['accuracy'])

    return model

We employed the deep learning library Keras to resolve the task of recognizing traffic signs. The dataset from GTSRB was used for training and testing [6]. The sixteen most common types of traffic signals can be categorized using the new system.

In order to construct a network architecture, there are several rules. Despite this, the majority of the network architecture design process is heuristic. Layers are chosen so that when they are added, data dimensionality decreases. However, there are no guidelines on specific layer macro settings.

Data volume and network depth should be correlated. Large networks and little data are likely to result in overfitted models. However, a shallow network with a lot of data would not provide adequate accuracy.

Table 1 describes the first developed network architecture. The architecture consists of several convolutional layers, maxPooling2D layer. All convolutional layers have a parameter stride equal to 2. This parameter determines the stride of the convolution sliding window, so layers with a parameter stride greater than 1 also combine the pooling operation.

Table 1. Neural network architecture.

|  |
| --- |
| Layer 1 |
| Convolutional, stride 2, kernel 5x5x60 |
| Convolutional, stride 2, kernel 5x5x60 |
| MaxPooling2D, pool\_size 2x2 |

When training a network proposed in Table 1 architecture, the classification accuracy reached a value of more than 0.9. However, this architecture seems to be exceeded due to a large number of layers. Thus, I decided to reduce the number of convolutional layers, which after several unsuccessful attempts resulted in the architecture, and add a Dropout layer presented in Table 2.

Table 2. Second neural network architecture.

|  |
| --- |
| Layer 2 |
| Convolutional, stride 2, kernel 3x3x30 |
| Convolutional, stride 2, kernel 3x3x30 |
| MaxPooling2D, pool\_size 2x2 |
| Dropout, 0.5 |

Finally, Table 3 shows the modified architecture of CNN. The reason why I add one more Dropout layer to reduce overfit while training and my model totally has 20 epochs.

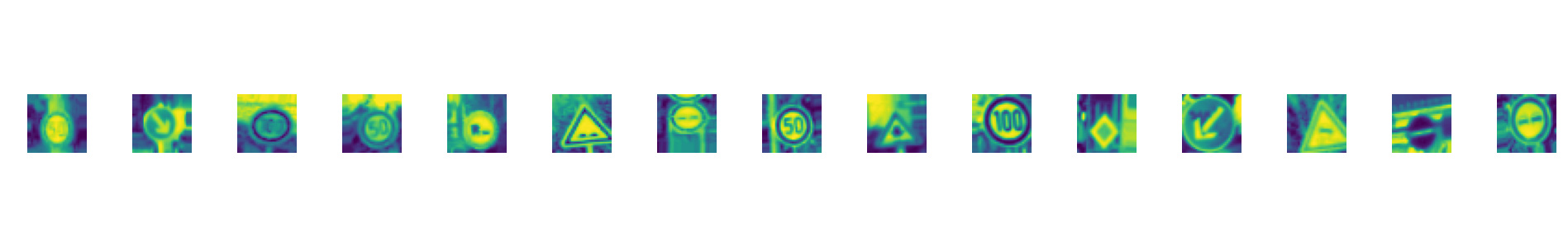
Table 3. Final neural network architecture.

|  |
| --- |
| Layer 2 |
| Fully connected -512 |
| Dropout, 0.5 |
| Softmax |

# **4. Traffic Sign Localization and Tracking**

The developed technology for traffic signs recognition consists of three steps: image preprocessing, localization and classification.

During image preprocessing, the input image’s color space is transferred to gray by the CV2 library Due to errors in the process of the image acquiring and the presence of small colored objects, some point-like noise occurs in the images after applying a threshold filter.

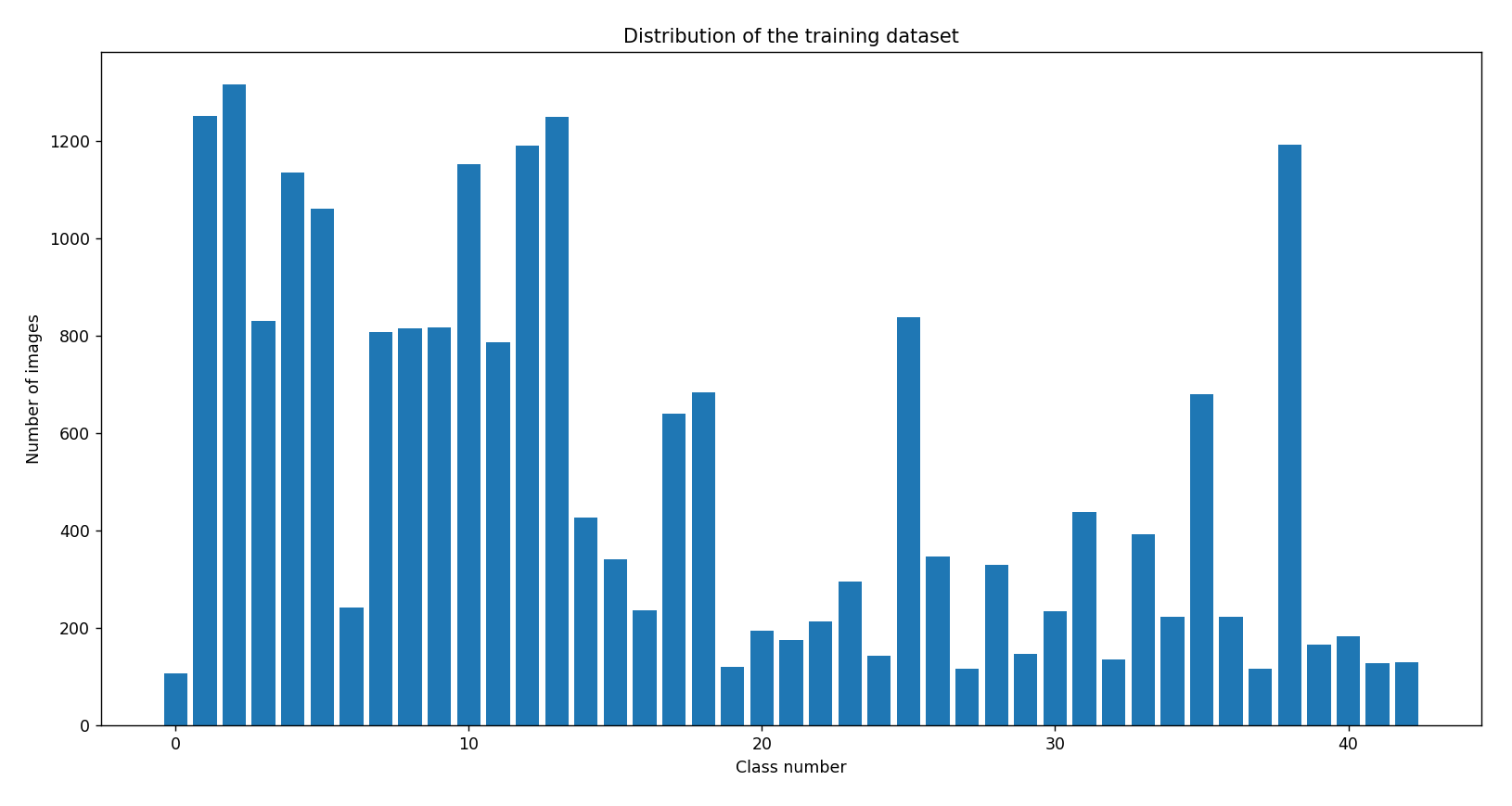


*Fig.7. Images of preprocessing stage*

The algorithms for identifying and following traffic signs are covered in Paper [5]. The time constraints for processing a single frame have been taken into consideration when developing the localization method, which is a modification of the generalized Hough transform. The algorithm performs well with the preprocessed images and produces useful results. The system's performance has increased as a result of tracking using the vehicle's current speed value because it allows for a significant reduction in the search area in subsequent frames. Additionally, the confidence in accurate recognition is significantly increased by the presence of a sign in the sequence of adjacent frames in predicted areas. The final step, classification, verifies that the entire process was carried out successfully.

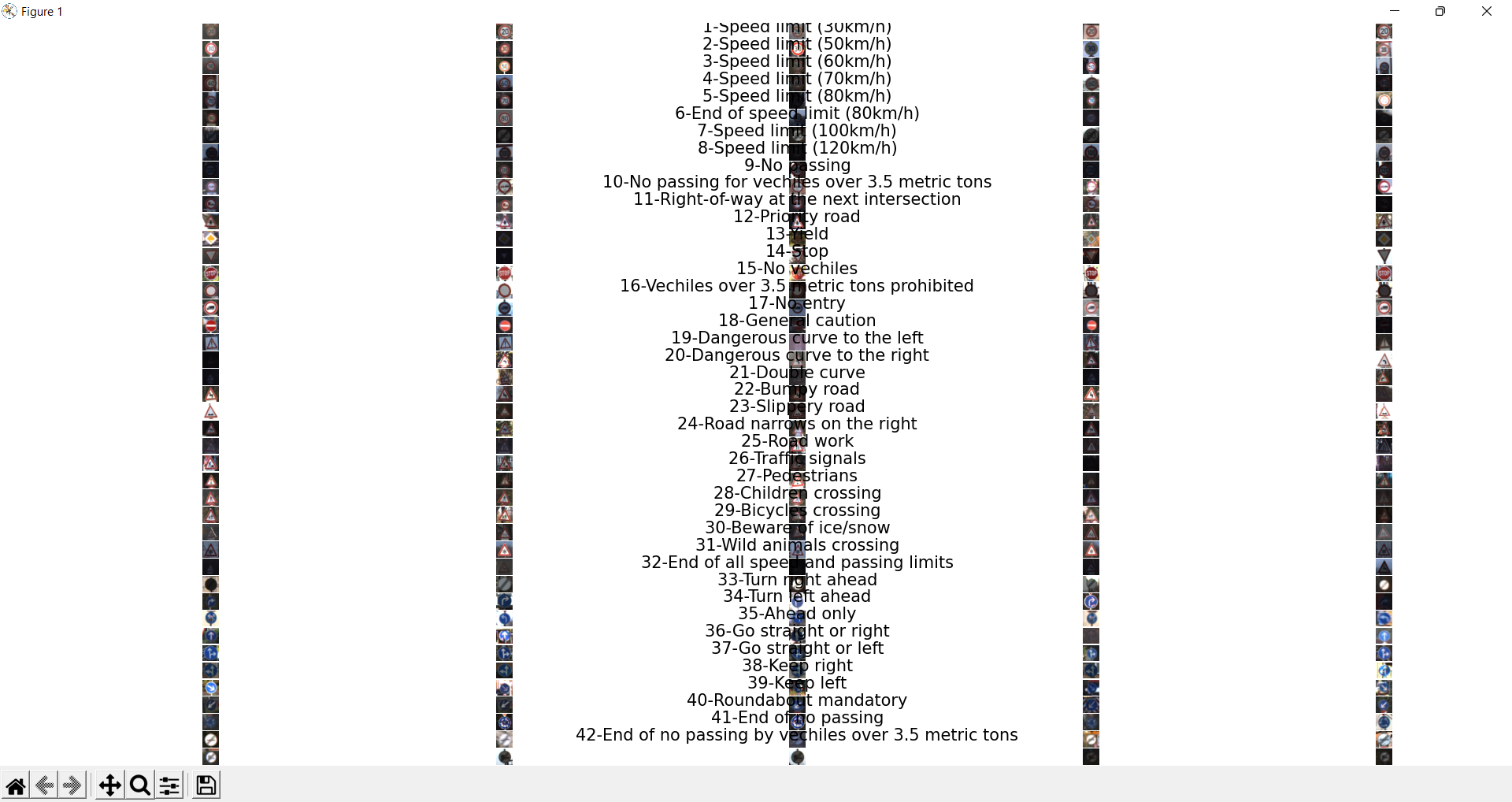
# **5. Training and Experimental Results**

To train and evaluate the model, the initial dataset was divided into the train and test datasets with the ratio 80/20 correspondently. More details, the Training dataset has a total of 22271 images with the size (32x32x3) and Validation has 5568 images with the size (32x32x3) and Test dataset has 6960 images with the size (32x32x3). My datasets are obtained from the GTSRB [6] and GTSDB [7] datasets and then are divided into the 43 most popular traffic signs. Fig.8 presents the distribution of the number of each class to the training dataset.



*Fig.8. Distribution of training dataset*

Besides, I also demonstrate all of the 43 most popular traffic signs and what each class is in Fig.9.



*Fig.9. Detail of each class*

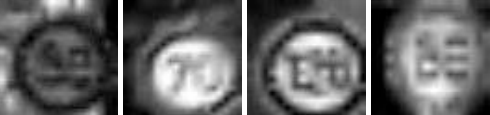
After preparing all of the necessary datasets for training I used my laptop ( AND Ryzen 5000 series) and it took me more than 1 hour to obtain the model. The training process ended shows results reaching precision equal to 99.89% when detecting a sign and 93.55% when classifying it.

The traffic sign images in Fig. 10 were effectively recognized by the CNN implementation suggested in this paper. The image demonstrates that the technology used produces good recognition results even with photographs of traffic signs, which are challenging for a person to recognize.



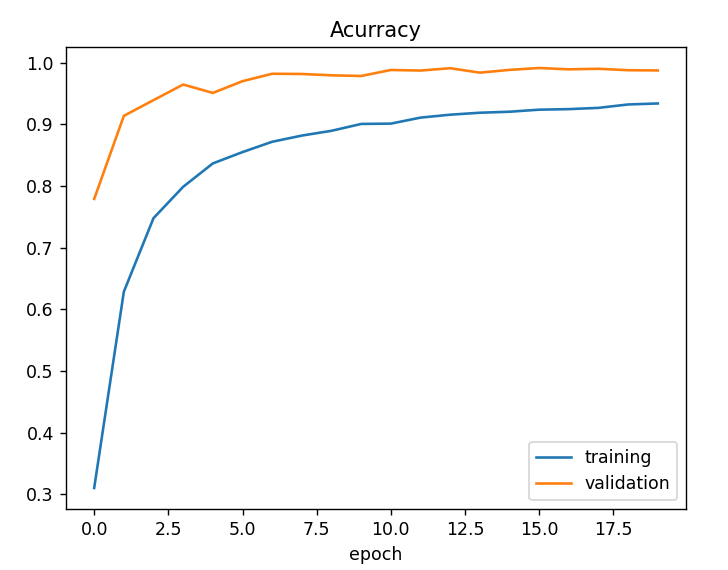
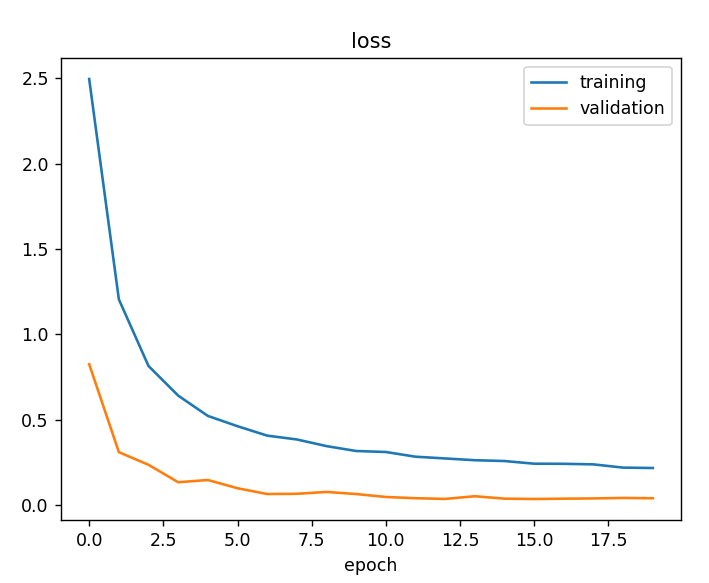
*Fig. 10. Successful classification*

However, the accuracy doesn’t reach 100 %. Fig. 11 shows the images of traffic signs that were recognized incorrectly



*Fig. 11. Unsuccessful classification*

Fig.12 also illustrates loss and accuracy during training with 20 epochs.



*Fig.12. Loss and Accuracy*

# **6. Conclusions**

The classification algorithm's implementation for the task of recognizing traffic signs is taken into consideration in this work.The suggested method for traffic sign classification exhibits very good results: 93.55% of correctly categorized photos when combined with preprocessing and localization procedures from other publications.

The use of our TSR algorithms allows the processing of video streams in real-time with CV2, and therefore at greater distances and with better quality than similar TSR systems have. FullHD resolution makes it possible to detect and recognize a traffic sign at a distance up to 50 m.

In future research, we plan to train the CNN to consider more traffic sign classes and possible bad weather conditions. Also, we plan to use a CNN not only for classification but for object detection too.

**References**

[1] A. Nikonorov, P. Yakimov, M. Petrov, Traffic sign detection on GPU using color shape regular expressions, VISIGRAPP IMTA-4, Paper 8 (2013).

[2] R. Belaroussi, P. Foucher, J.P. Tarel, B. Soheilian, P. Charbonnier, N. Paparoditis, Road Sign Detection in Images, A Case Study, 20th International Conference on Pattern Recognition (ICPR), 2010, pp. 484-488.

[3] A. Ruta, F. Porikli, Y. Li, S. Watanabe, H. Kage, K. Sumi, A New Approach for In-Vehicle Camea Traffic Sign Detection and Recognition,IAPR Conference on Machine Vision Applications (MVA), Session 15: Machine Vision for Transportation, 2009.

[4] V. Fursov, S. Bibkov, P. Yakimov, Localization of objects contours with different scales in images using Hough transform [in Russian],Computer Optics. 37, 4 (2013) 502-508.

[5] P. Yakimov, Tracking traffic signs in video sequences based on a vehicle velocity [in Russian], Computer Optics. 39, 5 (2015) 795-800.

[6] J. Stallkamp, M. Schlipsing, J. Salmen, C. Igel, Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition,Neural networks. 32 (2012) 323-332.

[7] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, C. Igel, Detection of Traffic Signs in Real-World Images: The {G}erman {T}raffic {S}ign {D}etection {B}enchmark, in: Proc. International Joint Conference on Neural Networks, 2013.

[8] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, S. Hu, Traffic-Sign Detection and Classification in the Wild. Proceedings of CVPR, 2016, рp. 2110-2118.

[9] Y. LeCun, P. Sermanet, Traffic Sign Recognition with Multi-Scale Convolutional Networks, Proceedings of International Joint Conference on Neural Networks (IJCNN'11), 2011.

[10] P. Yakimov, Preprocessing of digital images in systems of location and recognition of road signs [in Russian], Computer Optics. 37, 3(2013) 401-405.

[11] V. Terehov, D. Efimov, I. Tiukin, Neural network control system [in Russian], Textbook for high schools, High school, 2002.

[12] Djork-Arné Clevert, Thomas Unterthiner, Sepp Hochreiter, Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs), 2015.