Development of a Counterfeit Vehicle License Plate Detection System by Using Deep Learning

Burak Ağgül and Gökhan Erdemir

Abstract— **In this study, a deep learning-based counterfeit plate detection system that compares and detects vehicles with the make, model, color, and license plate is designed. As known that the relevant government institutions are responsible for keeping all detailed information about all motor vehicles in their database. All registration details are stored in the database. It is possible to find unregistered vehicles by comparing database records with detected details. In general, vehicles with counterfeit license plates are used in illegal actions. Therefore, it is of great importance to detect them. Generally, license plate recognition systems successfully detect counterfeit license plates that are randomly generated. Security units typically use such systems at toll roads, bridge crossings, parking lot entrances and exits, sites, customs gates, etc. This kind of system only checks the plate is exists or not in the database. But it is unsuccessful if the vehicle uses existing plate numbers such as stolen ones. In this study, the developed system can detect not only vehicles' plate numbers but also make, model, year, and color information by using deep learning. Thus, the system can also detect randomly generated plates and stolen plates that belong to another vehicle.**

Index Terms— **deep learning, convolutional neural networks (CNN), counterfeit plate.**

I. Introduction

DEEP LEARNING is a machine learning method that refers to deep neural networks. It is a system that allows us to train artificial intelligence to predict the outputs of a given dataset using artificial neural networks (ANN). Deep learning is a more advanced form of machine learning and ANN. The main difference is that it creates new features by learning from the data itself. While certain parts need to be defined and created correctly by users in machine learning, in deep learning, the system is expected to produce and label features and eventually produce an output using ANN [1]. In this study, a system that recognizes specific vehicle brand models, license plate information, and color information has been created. The system's primary purpose is to find the make and model information of the vehicle by using deep learning methods. The working principle of such systems, which can be added in positive or negative cases, is based on certain features of the vehicles. The vehicle's model, year, color and license plate, etc., distinguishing features are used. Today, it is seen that deep learning methods are used to get fast and accurate results from such applications. There are about 20 classes in deep learning methods. But these classes are methods that are derived from six categories. If it is necessary to give an example about derivation, it can be said to create a more complex structure by increasing the number of layers and recognizing this as a new approach. These six methods are (a) Convolutional Neural Networks (CNN), (b) Recurrent Neural Network (RNN), (c) Long/Short Term Memory (LSTM), (d) Restricted Boltzmann Machines (RBM), (e) Deep Belief Networks (DBN), and (f) Denoising Autoencoders (DAE). CNN is an algorithm that allows us to analyze various objects in data. CNN, a forward-looking neural network, is a method developed by considering the visual centers of animals as output. By convolutional processes, it can be regarded as a neuron's response to stimuli from its stimulus field [2-5]. RNN, with a simple example to understand the structure of recurrent neural networks more efficiently, puts our thoughts based on the knowledge we have learned before. In other words, when we read a sentence, we make sense of each word by considering the previous word. Traditional neural networks don't work that way. So this is a considerable shortcoming. RNN, on the other hand, has a structure that deals with such problems [1, 6]. LSTM networks are different from RNN networks. It has emerged as a structure that meets the need to estimate context gaps, known as the disadvantage of RNNs. [1, 7]. RBM is a neural network that can learn probabilistic distributions of Boltzmann machine input data set, which performs classification, regression, and feature learning [1]. DBN is created by stacks of Restricted Boltzmann Machines (RBM). RBMs are realized by training and learning, respectively. But horizontally, there is no connection between the layers [8, 9]. Autoencoders are a type of ANN commonly used for feature selection and extraction.

According to the structure of networks, the hidden layer can have more layers than the input layer [9, 10]. There is a risk for an identity function called the null function. It means that output equals the input, somehow. This situation makes Autoencoder useless. DAE can solve the problem by randomly turning some of the input values to zero. In general, the ratio of the input values which turn to zero should be set between

BURAK AĞGÜL, is with Computer Technologies Department, Istanbul Ayvansaray University, Istanbul, Turkey, (burakaggul@ayvansaray.edu.tr).
https://orcid.org/0000-0002-9183-1568

GÖKHAN ERDEMİR, is with Department of Electrical and Electronics Engineering, Istanbul Sabahattin Zaim University, Istanbul, Turkey, (gokhan.erdemir@izu.edu.tr).
https://orcid.org/0000-0003-4095-6333

Manuscript received March 25, 2022; accepted May 25, 2022.
DOI: 10.17694/bajece.1093158

30-50% depending on the data type [9, 10].

This study used CNN to detect counterfeit vehicle plates by considering vehicle registration information as the make, model, color, etc. A vehicle image dataset that has been generated by data augmentation was used for the training process [11]. In section II, CNN, one of the typical deep learning methods, is mentioned. This section also presents database structure, data augmentation, color, and license plate recognition steps. Section III presents the application steps of the system and its performance. The last section discusses the accuracy of the applied methods and the results we obtained.

II. Materials and Methods

We used the Python programming language and its deep learning libraries in this study. The system's primary purpose is that a make, model, etc., information of the vehicle as output is produced based on the trained data set results on the vehicle image given as input. The flow chart of the developed system is shown in the Figure 1.

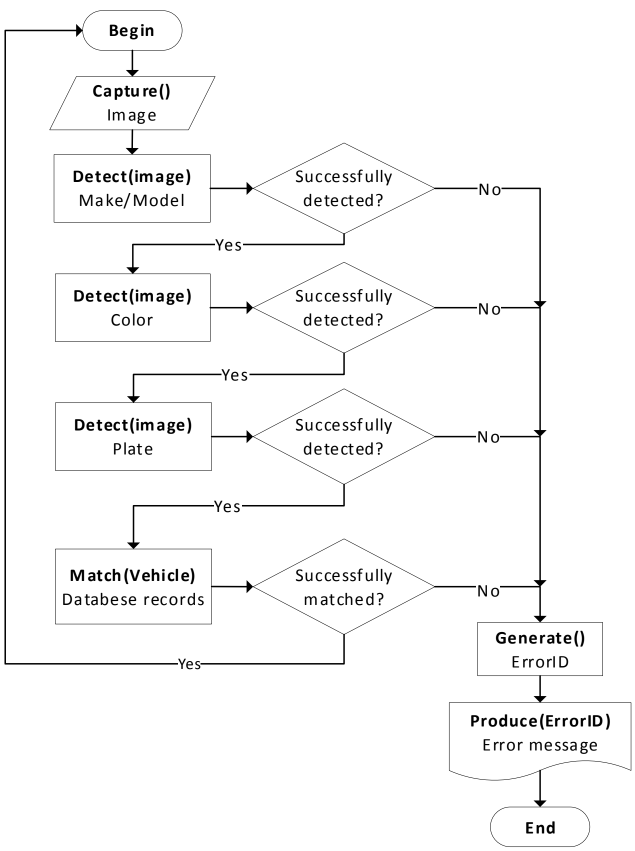

Fig.1. The flow chart of the system.

The first step of the system is to detect the vehicle's make and model on captured images from the video stream by using CNN. In this step, the dataset described in [11] was used for training. After detecting the make and the model of the vehicle, the vehicle color is recognized by using morphological operations on the image. And then, the vehicle plate number is recognized on the image by using pattern and character recognition techniques. The obtained information about the detected vehicle such as the make, model, year, color, and plate number is compared with database records as a final step. In this study, a virtual temporary database was generated to compare detected and actual information about the vehicle. If any errors occur during each detection step, an individual error code is generated by the system. The process is terminated by a warning message which includes the error code to the user. If an unregistered or suspicious vehicle is detected, such as make and model do not match with records, the system marks the detected vehicle. It produces a warning message for the user.

A. *Dataset*

The dataset's content consists of 10000 images for training, 1000 for testing, and 100 images for experimental studies [11]. All images in the dataset are in RGB format with 600x450 pixel resolution [11]. Figure 2 shows some examples in the dataset [11]. The data set was created as four classes based on the front and rear views of a single model of two brands. In the dataset [11], the Honda Civic and Ford Focus models were used. This study used the same brands and models denoted as HC and FF on the system, respectively. It is possible to increase brands and model variations by following the same steps [11].

Fig.2. Some examples in the dataset

B. *Data Augmentation*

The essential requirement for obtaining high performance and low error results from deep learning algorithms is the use of large amounts of data in the model's training process. In case of insufficient data, generating synthetic data using data augmentation methods is one of the most common approaches.

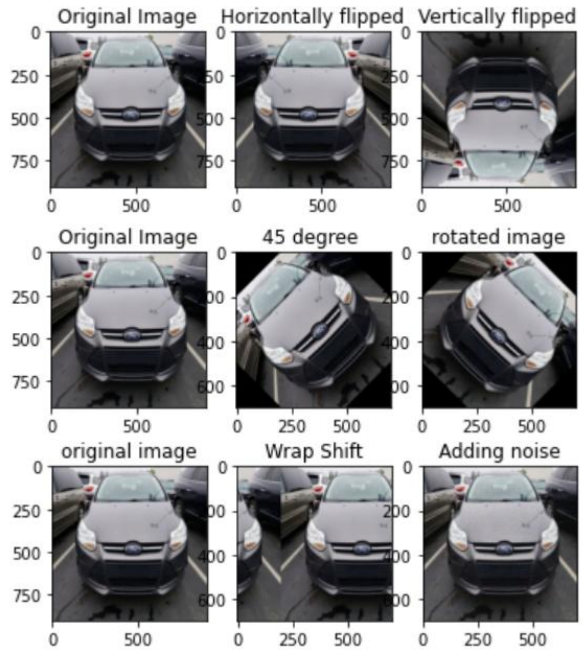

Fig.3. Samples for data augmentation

This study used data augmentation methods such as rotating, vertically shifting, horizontally symmetrical, vertically symmetrical, zooming, cutting, and adding noise to enrich the dataset [11]. Samples after data augmentation can be seen in Figure 3.

C. Convolutional Neural Networks (CNN)

In this study, we used CNN for training the model. To be explained most simply, this method is a deep learning algorithm that allows us to distinguish various objects or objects on an image from each other. Essentially, CNN consists of one or more convolutional layers, a subsampling layer, and one or more connected layers, a standard multilayer neural network. CNN solves the classification result using a typical neural network. But it also needs other layers to detect some information and features. The Convolutional Layer (CL) is the first layer of CNN. It detects low and high-level features on the image by applying some filters. In this layer, the image comes as input and is passed through a filter. It creates an attribute map due to the values formed after filtering. This stage can be seen in Figure 4 step by step. First, the filter is positioned in the upper left corner of the image. Then the indices of these two matrices are multiplied with each other, and all the results are added. After that, the filter is shifted to the right by 1 pixel, and the multiplication and addition processes are repeated. The same operations are applied to all pixels. The feature map is ready as a result of these operations. If the colored images use for classification, these processes are repeated for the RGB layers. Features are detected by moving the filter over the image and using matrix multiplication. More than one convolutional layer can be used on an image.The nonlinear layer, called the activation layer usually comes after the convolutional layers. The linearity is the main problem for this layer. While nonlinear functions such as sigmoid and then were used frequently in the past, today, the rectified linear activation function (ReLu) is commonly used that gives the best results in training the neural network. ReLu is usually used in the hidden layer, not the output. According to the problem, sigmoid or Softmax is preferred for the output [12]. The ReLu activation function is used to determine the best results. Pooling is generally used after the sequential convolutional layers. There are no learned parameters in this layer of the network. The goal is to minimize the computational complexity by keeping the number of channels of the input matrix constant and reducing the height and width information.

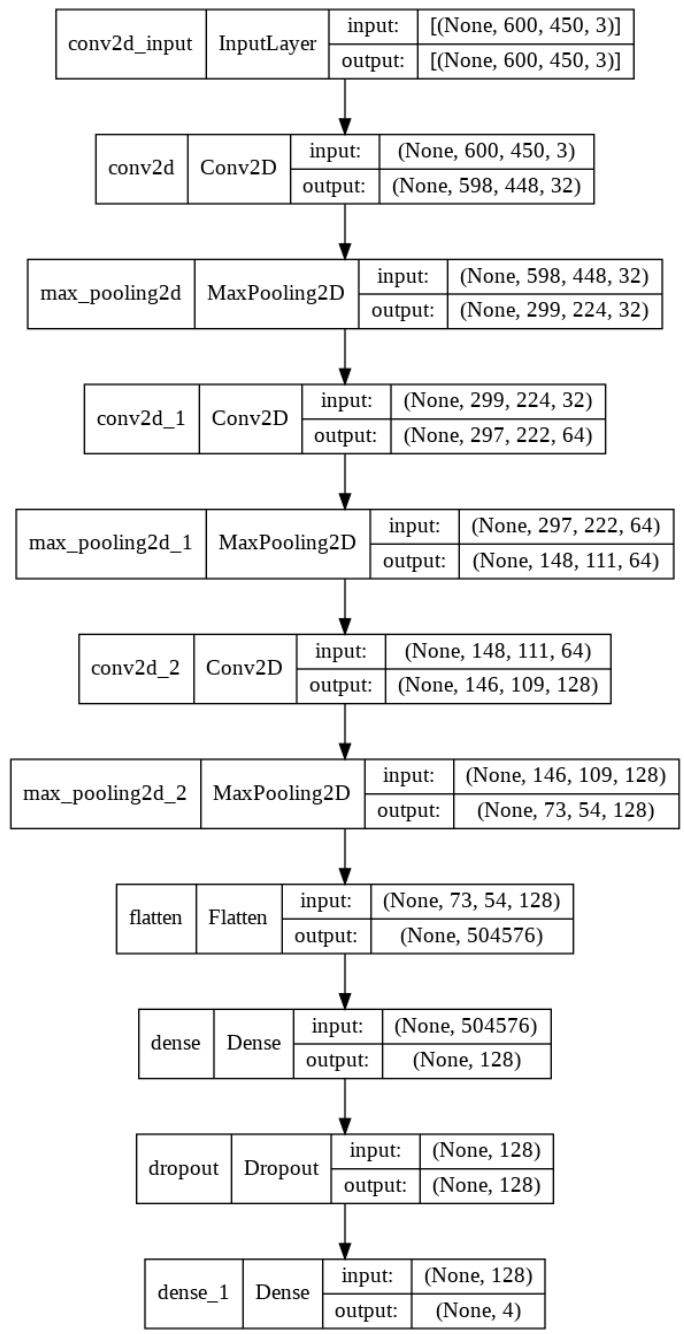

Fig.4. Designed CNN model's structure

The flattening layer is responsible for preparing data for the last layer, the fully connected layer. This layer contains the data converted into a one-dimensional array from the convolutional and the pooling layers. The fully connected layer's inputs are matrices created by the flattening layer. It is known as the result layer. It takes the value from the flattening output of the ANN as the input layer. Then, this layer produces the output value according to the activation function of the input values and the values it receives from the hidden layers. As a result of this process, the result estimation is produced with the Softmax activation function that has been determined. Another parameter used in the algorithm is Dropout, which is used in the fully connected layer. This parameter allows some neurons to be neglected to prevent overfitting during training [13].

Activation functions are used to introduce nonlinear features in ANN. In equation (1) for a simple neural network, (x) is used for the inputs, (w) for the weights, and (b) for the deviation/trend value. Output (y) is obtained by adding the activation function (f) to the output to form the result.

$$y = f\left(\sum_{i}^{n}(w_i x_i + b)\right) \tag{1}$$

ANN without an activation function acts as a linear function. Therefore, the neural network remains with limited learning power consisting of single-order polynomials. The main goal here is to learn what is desired in non-linear situations in the artificial neural network. The main reason is that actual data such as images, video, text, and audio will be given to our neural network to learn. In this way, our neural network performs faster and more accurately learning. On the other hand, the backpropagation algorithm is used to calculate the weights. The most essential strategy is to minimize the error rate. This is a step in the form of multiplying the inputs and the weights as shown in equation (1), sum with the bias value that shifts the activation function to the right or left, and applying the activation [14, 15].

Our model sequentially 32, 64, 128 units (128 outputs) with ReLu activated layers. For this reason, the ReLU function with 128 outputs are used while creating the fully connected layer. As shown in equations (2) and (3), it takes values in the range of ReLU [0, +∞).

$$f_1(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{2}$$

$$f_2(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases} \tag{3}$$

In general, in ANN models, low computational costs and short processing time are desired via aimed that more efficient results with low processing load. The ReLU is exactly at this point where it takes the value 0 for values on the negative axis [16-18] and decreases computational costs.

Four classes exist in the proposed model. Therefore, the model must produce four output values. Adding a new layer can suffice to solve this problem. However, before adding the new layer, the dropout value is set as 0.5 to reduce the weights by half and prevent over-learning. Then, a new layer with four outputs with a softmax activation function adds to the model.

The softmax function structure is similar to the sigmoid function. It is used in the output layer of the model when there are more than two classification problems [12]. As shown in Equations (4) and (5), it provides a probability value in the range of 0-1 to determine whether the input belongs to a specific class or not [17, 19].

$$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{i=1}^{j} e^{x_j}}, i = 1,2, \ldots j \tag{4}$$

$$\frac{\partial f_i(\vec{x})}{\partial x_j} = f_i(\vec{x})(\delta_{ij} - f_i(\vec{x}) \tag{5}$$

D. Color and License Plate Recognition

Vehicle color detection is a subsystem that estimates a vehicle's color by performing morphological operations on the image [20]. For this process, RGB values of all pixels on the image cropped from certain regions are gathered, as shown in Figure 5 on the picture of the vehicle. The most common RGB value is estimated as the color of the vehicle. The accuracy in the color estimation of the vehicle is directly related to the resolution quality of the image [20].

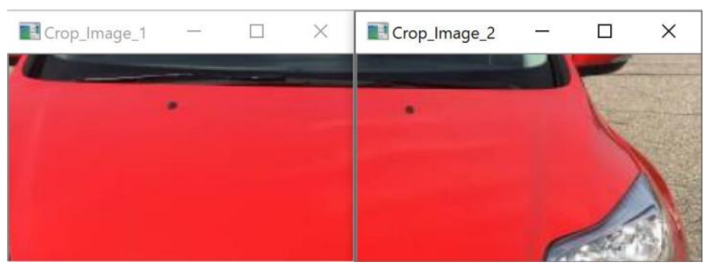

Fig.5. Cropped image to estimate vehicle's color**.**

License plate recognition systems are generally used for speed violations, light violations, toll highway crossings, customs, etc. The picture is taken with a camera from the front or rear. And then, the plate number area is detected by applying morphological processes [23]. The color image is converted to be entirely gray. Edge vertices on the image are determined by using an edge detection algorithm on the resulting image. The license plate number is converted from the image to the text using pattern and character recognition algorithms on the detected area [20-23].

III. Case Studies

Deep learning, image processing, and mathematics libraries of the Phyton are used during coding. On the other hand, TensorFlow, is an open-source deep-learning library, was used for creating the deep learning model. Its flexible structure allows you to deploy calculations using one or more CPUs, GPUs, with a single API, regardless of platform. The other essential library in this study is Keras, an open-source neural network library. Numpy library, a basic math library used for scientific calculations, was also used. PIL is a graphics processing library used to perform any operation on an image. Imutils library was used to flip, rotate, resize, contour extraction, edge detection, etc., on an image. PyTesseract, a character recognition tool that allows us to perform morphological operations, was used for the plate number recognition process. Besides, synthetic images were used to perform segmentation, geometric transformations, color space manipulation, analysis, filtering, morphology, feature detection on an image.

TABLE I
COMPARISON OF ACTIVATION FUNCTIONS

Activation functions	Training loss	Training accuracy	Test loss	Test accuracy	Epoch
sigmoid	1,3863	0,2498	1,3863	0,2500	10
tanh	1,3965	0,2498	1,3966	0,2500	10
ReLU	0,0939	0,9656	0,1360	0,9520	10
LeakyReLU	0,0673	0,9767	0,1054	0,9640	10
swish	1,3862	0,2498	1,3862	0,2500	10

While creating the proposed deep learning model, the model was compared with five different activation functions. The original dimensions of 600x450 pixels were entered in the size of the images in the database used for training and testing. It was performed on the Google Colab server, which provides approximately 12 hours of limited-time support by giving ten epochs and batch_size 32 values. In Table I, it can be seen that the ReLU and LeakyReLU activation functions gave the best results. In addition, the train and test values of the functions were quite close to each other. This is because the LeakyReLU function is derived from the ReLU function. Two activation functions were used to determine our model with an epoch value of 40. Table II and Table III show that the ReLU function is more successful with a slight value difference.

TABLE II
OPTIMIZED VALUES OF ReLU

Batch Size = 32		ReLU		
		adam	*rmsprop*	*sgd*
1 Epoch	loss	1,009	1,715	1,321
	accuracy	0.608	0.559	0.351
	val_loss	0.621	0.642	1,165
	val_ accuracy	0.741	0.727	0.499
10 Epoch	loss	0.138	0.268	0.461
	accuracy	0.949	0.922	0.808
	val_loss	0.111	0.167	0.436
	val_ accuracy	0.966	0.943	0.827
20 Epoch	loss	0.074	0.302	0.272
	accuracy	0.974	0.927	0.898
	val_loss	0.080	0.175	0.235
	val_ accuracy	0.985	0.958	0.919
30 Epoch	loss	0.059	0.378	0.186
	accuracy	0.981	0.910	0.928
	val_loss	0.038	0.287	0.183
	val_ accuracy	0.988	0.949	0.930
40 Epoch	loss	0.048	0.423	0.138
	accuracy	0.985	0.901	0.945
	val_loss	0.067	0.324	0.129
	val_ accuracy	0.987	0.955	0.962
Training and Test Results	Training Loss	0.013	0.152	0.067
	Training Acc.	0.996	0.958	0.975
	Test Loss	0.067	0.324	0.129
	Test Acc.	0.987	0.955	0.962

In this study, the test images used for a vehicle make/model and color detection consist of 25 pictures with 600x480 pixel dimensions. before the 16 vehicle images used for the comparison of vehicle brand, color, and license plate information were used, the plates on each vehicle image were created by using characters such as X, W, unlike the symbols used on the vehicle plates in the Republic of Turkey, within the scope of the Personal Data Protection Law.

TABLE III
OPTIMIZED VALUES OF LeakyReLU

Batch Size = 32		LeakyReLU		
		adam	*rmsprop*	*sgd*
1 Epoch	loss	1,789	1,356	1,321
	accuracy	0.476	0.326	0.351
	val_loss	0.694	12.57	1,165
	val_ accuracy	0.733	0.453	0.499
10 Epoch	loss	0.320	0.398	0.461
	accuracy	0.910	0.838	0.808
	val_loss	0.295	0.378	0.436
	val_ accuracy	0.931	0.853	0.827
20 Epoch	loss	0.411	0.235	0.272
	accuracy	0.907	0.907	0.898
	val_loss	0.320	0.201	0.235
	val_ accuracy	0.964	0.932	0.919
30 Epoch	loss	0.459	0.159	0.186
	accuracy	0.892	0.937	0.928
	val_loss	0.377	0.152	0.183
	val_ accuracy	0.957	0.941	0.930
40 Epoch	loss	1,068	0.121	0.138
	accuracy	0.882	0.953	0.945
	val_loss	0.643	0.103	0.129
	val_ accuracy	0.936	0.963	0.962
Training and Test Results	Training Loss	0.206	0.051	0.067
	Training Acc.	0.936	0.983	0.975
	Test Loss	0.643	0.103	0.129
	Test Acc.	0.936	0.963	0.962

The vehicle information from which the brand/model, license plate, and color information was obtained were compared with the records in the database. As shown in Table IV, it is defined within the records that when the vehicle's make/model, license plate, and color information match, it sends a successful message to the screen. If any information does not match, it gives a failed message. While the resolution values for the images were minimum of 500x500 pixels, successful results were obtained in 14 images with an accuracy of 87.5% for 16 test images and a failure rate of 12.5% in 2 images due to the low resolution of the test. When testing images with low pixel resolution, it was observed that they failed in color detection, plate region detection, or character reading.

TABLE IV
TEST RESULTS FOR 100 RANDOMLY CHOSEN IMAGES

Make / Model	Images number	Accurate Detection	Accuracy rate
2012_2014_FFF	25	25	100%
2012_2014_FFR	25	25	100%
2016_2019_HCF	25	25	100%
2016_2019_HCR	25	25	100%

IV. CONCLUSION

According to training and test results which were shown in Table II and III, it can be seen that the developed system almost has 98.7% accuracy on the test data to detect a vehicle's brand and model. The color recognition subsystem was designed as the sum and average of the pixel numbers on the image pieces taken by giving specific coordinates on the vehicle image without being subject to learning, training, or testing. The most significant handicap for this process, a simple and straightforward method, is the reflection of light in sunny weather. It is thought that the system will work with the same performance when a larger and more complex data set is used. As a result of this study, a make/model, year, and color recognition system has been developed that can be integrated into the existing license plate recognition systems.

REFERENCES

[1] A. Şeker, B. Diri, and H. Balık, “Derin Öğrenme Yöntemleri ve Uygulamaları Hakkında Bir İnceleme”, Gazi Journal of Engineering Sciences, vol. 3, no. 3, 2017, pp. 47–64.
[2] K. Fukushima, “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position”, Biological Cybernetics, vol. 36, no. 4, 1980, pp. 193–202.
[3] Z. Yumeng, C. Peng, F. Liuping and C. Fangfang, "Research on Pseudo-Random Noise Information Identification Technology of Printed Anti-Counterfeiting Image Based on Deep Learning", 2020 5th International Conference on Computer and Communication Systems (ICCCS), 2020, pp. 206-209.
[4] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-Based Learning Applied to Document Recognition”, Proc. of the IEEE, vol. 86, no. 11, 1998, pp. 2278-2324.
[5] Y. Lecun, L.D. Jackel, B. Boser, J.S. Denker, H.P. Graf, I. Guyon. D. Henderson, R. E. Howard and W. Hubbard, “Handwritten digit recognition: applications of neural network chips and automatic learning”, IEEE Communications Magazine, vol. 27, no. 11, 1989, pp. 41-46.
[6] J. L. Elman, “Finding structure in time”, Cognitive Science, vol. 14, no. 2, 1990, pp. 179–211.
[7] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory”, Neural Computation, vol. 9, no. 8, 1997, pp. 1735–1780.
[8] G. E. Hinton, “Reducing the Dimensionality of Data with Neural Networks”, International Encyclopedia of Education, vol. 313, no. July, 2006, pp. 468–474.
[9] L. Tang, Z. Gao and L. Huang, "Plate Recognition Based on Deep Learning", 2018 12th IEEE International Conference on Anti-counterfeiting Security and Identification (ASID), 2018, pp. 116-120.
[10] Ş. Abdulkadir and A. G. Yüksek, “Stacked Autoencoder Method for Fabric Defect Detection”, Cumhuriyet Science Journal, vol. 38, no. 2, 2017, pp. 342–342.
[11] G. Erdemir and B. Ağgül, “Data Augmentation for a Learning-Based Vehicle Make-Model and License Plate Matching System”, European Journal of Technic, vol. 10, no. 2, 2020, pp. 331–339.
[12] C. Bircanoğlu and N. Arıca, "A comparison of activation functions in artificial neural networks", 2018 26th Signal Processing and Communications Applications Conference (SIU), 2018, pp. 1-4.
[13] N. Srivastava and G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 15, 2018, pp. 7642–7651.
[14] M. M. Lau and K. H. Lim, “Review of adaptive activation function in deep neural network”, 2018 IEEE EMBS Conference on Biomedical Engineering and Sciences, IECBES 2018 - Proceedings, 2019, pp. 686–690.
[15] H. H. Tan and K. H. Lim, “Vanishing Gradient Mitigation with Deep Learning Neural Network Optimization”, 2019 7th International Conference on Smart Computing and Communications (ICSCC), 2019, pp. 7–10.
[16] M. Kaloev and G. Krastev, "Comparative Analysis of Activation Functions Used in the Hidden Layers of Deep Neural Networks", 2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), 2021, pp. 1-5.
[17] O. Sharma, “A New Activation Function for Deep Neural Network”, Proceedings of the International Conference on Machine Learning, Big Data, Cloud and Parallel Computing: Trends, Prespectives and Prospects, COMITCon, 2019, pp. 84–86.
[18] H. Chung, S. J. Lee, and J. G. Park, “Deep neural network using trainable activation functions”, Proceedings of the International Joint Conference on Neural Networks, vol. 2016-October, no. l, pp. 348–352.
[19] M. A. Mercioni and S. Holban, "The Most Used Activation Functions: Classic Versus Current", 2020 International Conference on Development and Application Systems (DAS), 2020, pp. 141-145.
[20] B. Ağgül and G. Erdemir, “Açık Kaynak Kodlu Taşıt Renk Tespit Yazılımı Geliştirilmesi”, İstanbul Sabahattin Zaim Üniversitesi Fen Bilimleri Enstitüsü Dergisi, vol. 3, no. 1, Mar. 2021, pp. 47-50.
[21] O. Kaplan, Ş. Sağıroğlu, Ö.F. Çolakoğlu, “Erciyes Üniversitesi Bilgisayar Mühendisliği Bölümü Araç Tanıma Sistemi”, 2002, pp. 2–6
[22] Ş. Sağıroğlu and E. Beşdok, “A Novel Approach for Image Denoising Based on Artificial Neural Networks”, vol. 15, no. 2, 2012, pp. 71–86.
[23] O. Bingöl and Ö. Kuşçu, “Bilgisayar Tabanlı Araç Plaka Tanıma Sistemi”, Bilişim Teknolojileri Dergisi, vol. 1, no. 3, 2008, p. 1-5.

BIOGRAPHIES

BURAK AĞGÜL received his undergraduate degree from Istanbul Sabahattin Zaim University, Faculty of Engineering and Natural Sciences, Department of Computer Engineering, in 2015. He graduated from Istanbul Sabahattin Zaim University Graduate Education Institute, Department of Computer Science and Engineering in 2020. Now, he is a Ph.D. student in the same department. He works as a lecturer in Istanbul Ayvansaray University Plato Vocational School Computer Technologies Department. His areas of interest are artificial intelligence, deep learning, machine learning, and image processing.

GÖKHAN ERDEMİR received his B.Sc., M.Sc. and Ph.D. degrees from Marmara University, Turkey, respectively. During his Ph.D., we worked as a research scholar at Michigan State University, Department of Electrical and Computer in East Lansing MI, USA. Now, he is an assistant professor at Istanbul Sabahattin Zaim University, Department of Electrical and Electronics Engineering. His research topics include robotics, mobile robotics, and control systems.