Vehicle Make Detection Using the Transfer Learning Approach

Dovile Komolovaite[1, 3, *], Andrius Krisciunas[1], Ingrida Lagzdinyte-Budnike[1], Aurelijus Budnikas[2], Dominykas Rentelis[3]
[1]Department of Applied Informatics, Kaunas University of Technology, Studentu g. 50–407, LT-51368 Kaunas, Lithuania
[2]Department of Software Engineering, Kaunas University of Technology, Studentu g. 50-406, LT-51368 Kaunas, Lithuania
[3]MB "Kodo technologijos", Silagirio St. 73-1, Vijukai, LT-54306 Kaunas dist., Lithuania
dovile.komolovaite@ktu.edu

***Abstract*—Vehicle detection and classification is an important part of an intelligent transportation surveillance system. Although car detection is a trivial task for deep learning models, studies have shown that when vehicles are visible from different angles, more research is relevant for brand classification. Furthermore, each year, more than 30 new car models are released to the United States market alone, implying that the model needs to be updated with new classes, and the task becomes more complex over time. As a result, a transfer learning approach has been investigated that allows the retraining of a model with a small amount of data. This study proposes an efficient solution to develop an updatable local vehicle brand monitoring system. The proposed framework includes the dataset preparation, object detection, and a view-independent make classification model that has been tested using two efficient deep learning architectures, EfficientNetV2 and MobileNetV2. The model was trained on the dominant car brands in Lithuania and achieved 81.39 % accuracy in classifying 19 classes, using 400 to 500 images per class.**

***Index Terms*—Image classification; Machine learning; Vehicle detection.**

I. INTRODUCTION

The ability to recognize a vehicle from a video can be very helpful, from traffic monitoring to following the fleeing driver from a crime scene. So far, Automatic License Plate Recognition systems have been mostly utilized to identify the vehicle under investigation. However, if the criminal is driving fast and the recorded license plates are of poor quality, the main tool for locating the suspect is monitoring a car with certain attributes via city cameras and manual searches by officers. Nonetheless, suspects can swap license plates, or witnesses can write down only part of the license plate or recall only a few facts about the car, such as color, make, or model. As a result, an automatic vehicle characteristics recognition system becomes critical in assisting officers and improving the intelligent transportation system [1].

The potential of classifying key characteristics of a vehicle is constantly being examined due to the increasing use and research of convolutional neural networks (CNNs) and the development of the graphics processing unit. CNN has already been used to detect and track cars or to count the number of cars passing by on the road [2], [3]. However, due to the most severe CNN constraint of requiring a large dataset, the transfer learning approach [4] was designed to tackle limited data resources. Deep learning technology is already helping individuals to reduce eye tracking on multiple screens, so it can also be used to determine the attributes of a car, thus helping automate toll collection, intelligent parking systems [5] or law enforcement [6]. Filtering video segments by specific car attributes can be simplified using a vehicle classification model.

The task of identifying vehicle characteristics has unique challenges and problems. The first is the impact of weather, time of day, and varied lighting exposures, all of which can make the model difficult to perform, as the classes are very similar [1]. The lack of open-source datasets that include a range of commonly used vehicles, viewing angles, diverse image quality, and data with more than several images per class is another barrier [7]–[9]. And the third challenge is that newer vehicle models are released on a regular basis, requiring the deep learning model to be retrained with even small amounts of data. According to statistics, 42 new models were created in the United States alone in 2020, 34 in 2021, and 62 models are expected in 2025 [10], demonstrating an upward trend in car manufacturing. To address the last two challenges, a country-specific dataset is collected and vehicle brand classification is explored using a transfer learning technique that does not require a large dataset for training, making it appropriate for constantly upgrading systems.

The aim of this research is to create a retrainable vehicle detection and classification framework that can accurately classify vehicle images from various viewpoints to simulate the effect of city cameras. To present the most popular cars on the roads, the dataset of vehicle brands is taken from the Lithuanian car marketplace website. This research uses architectures designed for lower computing power,

Manuscript received 25 January, 2022; accepted 31 March, 2022.

MobileNetV2 and EfficientNetV2, to ensure that the solution can be implemented in real time and retrained with new vehicle models. The pre-trained feature vectors of defined architectures are fine-tuned, and the study examines multiple combinations of dense classification layers to suggest a novel architectural design. Finally, the generalization abilities of the model are evaluated by comparing the classification performance of various distribution datasets, and examples of the final proposed system are presented. At the end of the study, the findings are discussed, and conclusions and recommendations for further research are presented.

II. Computer Vision Literature Review

Convolutional Neural Networks are widely used in computer vision tasks to classify objects such as cars, pedestrians, and more [11]. To learn how to replicate human intellect, this supervised machine learning technique requires labeled data [12]. The basic CNN consists of convolutional, polling, and fully connected layers that classify the final classes, where the convolutional layer extracts low- and high-level visual features using a weight-sharing mechanism [11].

A state-of-the-art CNN model named "You Only Look Once" (YOLO) demonstrated exceptional performance in real-time image processing due to its high performance and accuracy [13]. The third version of the model is the latest and includes 53 convolutional layers and 23 residual layers. The YOLOv3 model has already been pre-trained on the COCO dataset, which covers 80 object categories, including the means of transportation such as bicycles, cars, buses, trains, trucks, and so on. Furthermore, it has been used successfully in vehicle detection applications and has improved recognition capabilities for tiny objects [13].

A. Transfer Learning

CNNs, on the other hand, have the disadvantage of being strongly reliant on large amounts of data to avoid overfitting [14] and hence requiring massive computer resources [15]. As a result, a machine learning technique called "transfer learning" [4] was established, in which a previously trained model with stored weights is used to train a new model with a smaller similar dataset. Because pre-trained models have already learned to recognize edges, colors, and patterns, the weights of the feature extraction network can be frozen to convey this information [16]. Finally, the last layers can be used to obtain additional detailed information about the classified classes [15]. Another transfer learning method is fine-tuning, which involves retraining the layers with initialized pre-trained model weights rather than freezing them [17]. In terms of time and computational resources, freezing layers is the most effective strategy, whereas training from scratch is the most expensive [18].

B. Classification of Vehicle Attributes

Vehicle classification is one of the most interesting tasks in an intelligent transportation system. It has been studied using a logo-based technique, in which the vehicle logo is first localized and then classification is performed [12]. Classification of car brands and models based on front images reaches accuracies of 98.22 % (49 classes) [5], 98.7 % (107 classes) [19], 97.89 % (35 classes) [1], and 96.33 % (766 classes) [20]. However, the region of interest is usually strongly cropped to remove the background and requires a certain angle image; this strategy, for example, is suited for parking lot cameras. There have been successful experiments to determine the type of vehicle, such as a car, van, truck, or another, with an accuracy of up to 99.68 % using a modified pre-trained ResNet-152 model [3] or 76.28 % using ResNet34 [4]. Another example shows that the AlexNet architecture can be used to produce a perfect classifier to assess the position of a vehicle [16]. This type of study can assist in the development of multiple models for different viewpoints to improve the overall performance of model recognition. The classification of vehicle brands using the logo has been of interest for many years, but the challenge of detecting cars that do not rely on this information only has remained largely unsolved [21].

Another type of vehicle classification is the appearance-based classification, which uses elements such as lights, windows, and the vehicle body to classify the vehicle [12]. To date, the most used car data from different viewpoints are the Stanford car dataset, which consists of 196 model classes with 24 to 68 images in the training dataset and the same amount of test data as it has been split 50/50 ratio [6], [15], [16], [18]. The publications that use these data do not take into consideration the impact of unbalanced data, although the sample size of the test data is almost three times smaller for certain classes. However, using this dataset, the fully trained GoogLeNet architecture [9] and the MobileNet transfer learning model [22] had the highest accuracies of the classification of the make and model, at 80 % and 78.27 %, respectively. In 2020, a unique study was carried out to actualize the classification of vehicle models using real surveillance cameras, and the model achieved an accuracy of 62.09 % in real environmental settings [23]. This, together with the increasing number of vehicles and city cameras [7], shows that vehicle classification is relevant and more research is needed to make it adaptive to real-world scenarios.

C. EfficientNet and MobileNet Architectures

This study selected two of the most efficient classification architectures. The first architecture is the lightweight MobileNetV2 [24], which, according to the reviewed articles, achieves one of the highest accuracies. It employs a slow downsampling strategy and more layers have large feature maps, resulting in detail preservation [25]. Another selected architecture is EfficientNetV2, a newer architecture network that uses a compound scaling approach to uniformly scale and balance the depth, width, and resolution of the network to improve accuracy and efficiency [26]. The MobileNetV2 architecture is pre-trained using the ImageNet dataset, which contains 1000 classes covering high-level categories such as animals and vehicles [27]. Meanwhile, EfficientNetV2 is pre-trained on ImageNet-21k, which has 21,841 classes, and is simply a larger version of ImageNet [27]. The pre-training on ImageNet-21k is claimed to provide better performance than the use of ImageNet1k [28].

III. PROPOSED VEHICLE MAKE DETECTION SYSTEM

The suggested system architecture is illustrated in Fig. 1. Here, the preparation phase involves the preparation of data and the training of the classifier, while the usage phase involves the adaptability of the model to the use of external applications containing traffic camera data. The final approach can locate the car in the frame, classify the make for each bounding box, and be leveraged for rendering or other purposes.

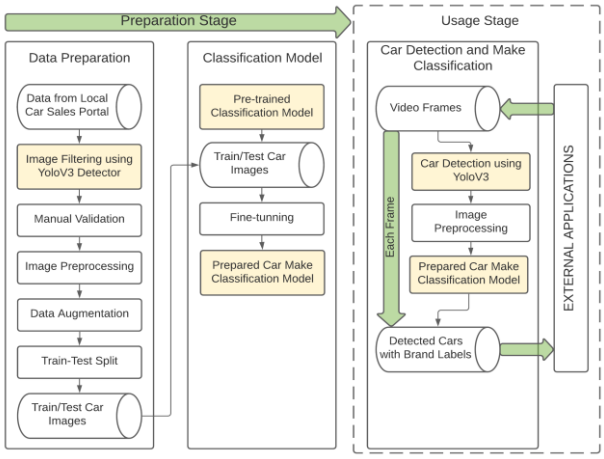


Fig. 1. The architecture of the proposed vehicle detection system.

Each of the three phases is described in further detail:

– *Data Preparation*. The source material contains images of vehicles for sale posted on the Internet. These images contain a lot of noise, such as the gearbox, steering wheel, engine, car interior, etc. The pre-trained YOLOv3 state-of-the-art detection model was utilized as an auto labeler to filter out unnecessary data and crop the region of interest. To ensure that no noisy data was left out, the photographs were reviewed using the designed interface to exclude unrepresentable images. There is also a problem with data-class imbalance; thus, only classes with at least 400 images are taken, and those with more than 500 images are truncated. Image pre-processing includes resizing, converting to grayscale, and normalizing such that the image input matches the pre-trained data. Subsequently, data augmentation was used to create a more diversified representation of the vehicle's sides and dimensions. The last step includes data separation training and testing.

– *Classification Model*. The classification task is the central objective of the study. The most efficient architectures of the pre-trained models, MobileNetV2 and EfficientNetV2, were investigated by fine-tuning their feature vectors and using different dense classifiers. With a limited dataset, these models could easily be retrained. See the subsection "Image Classification Using Various Classifiers" for further information.

– *Car Detection and Make Classification*. The YOLOv3 model, which has previously been used for vehicle image filtering, is combined with the best classifier. This can be used for vehicle monitoring via video cameras.

Image Classification Using Various Classifiers. The suggested classification architecture has two primary components: reusing a previously trained CNN model and adding a new classifier. For initial feature extraction, selected model architectures with pre-trained weights are used. The MobileNetV2 architecture was trained using a considerably larger and more general ImageNet dataset to obtain the vector of image features. With a total of 2257984 parameters, the vector output size is 1280. Meanwhile, the feature vector of the EfficientNetV2 neural network is trained on the ImageNet-21k dataset, resulting in a vector output size of 1536 and nearly 6 times more parameters: 12930622. Although the learning time increases, the feature vector is fine-tuned with new classifier layers to obtain greater accuracy. The classifier receives the resulting dense 1-D tensor feature vector output and learns to categorize the input image according to the brands of the vehicle. Figure 2 illustrates the classification architecture by reusing pre-trained feature vectors and adding new classifier layers.

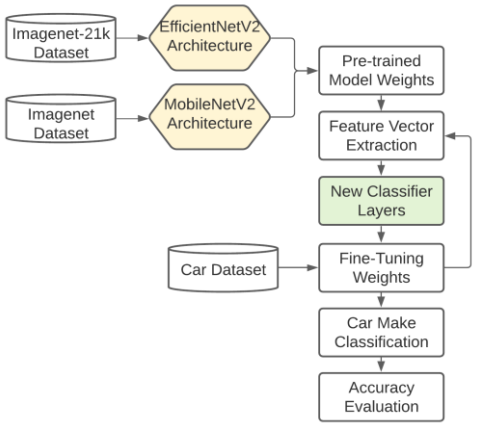


Fig. 2. Image classification using fine-tuning.

When the image feature vector is extracted using a grid with pre-trained weights rather than starting from a random weight initialization, then the generated classifier is based on a fully connected grid. In this way, the knowledge already acquired can be re-used instead of training from scratch. The classifier can be as simple as a dense layer with a Softmax activation function to obtain the probability of each class. This classification is called the "baseline" or "version 1". Different classifier versions are tested for greater accuracy. The 2nd version adds weight adjustment L1

and L2 to reduce the number of features. A batch normalization layer is added in the 3rd version to allow the network to train faster. Dropout at a rate of 0.5 is utilized in the 4th version to avoid overfitting and learn more robust features. And different combinations of dense layers are then introduced with the rectified linear block activation function. All classifier architectures are presented in Fig. 3.

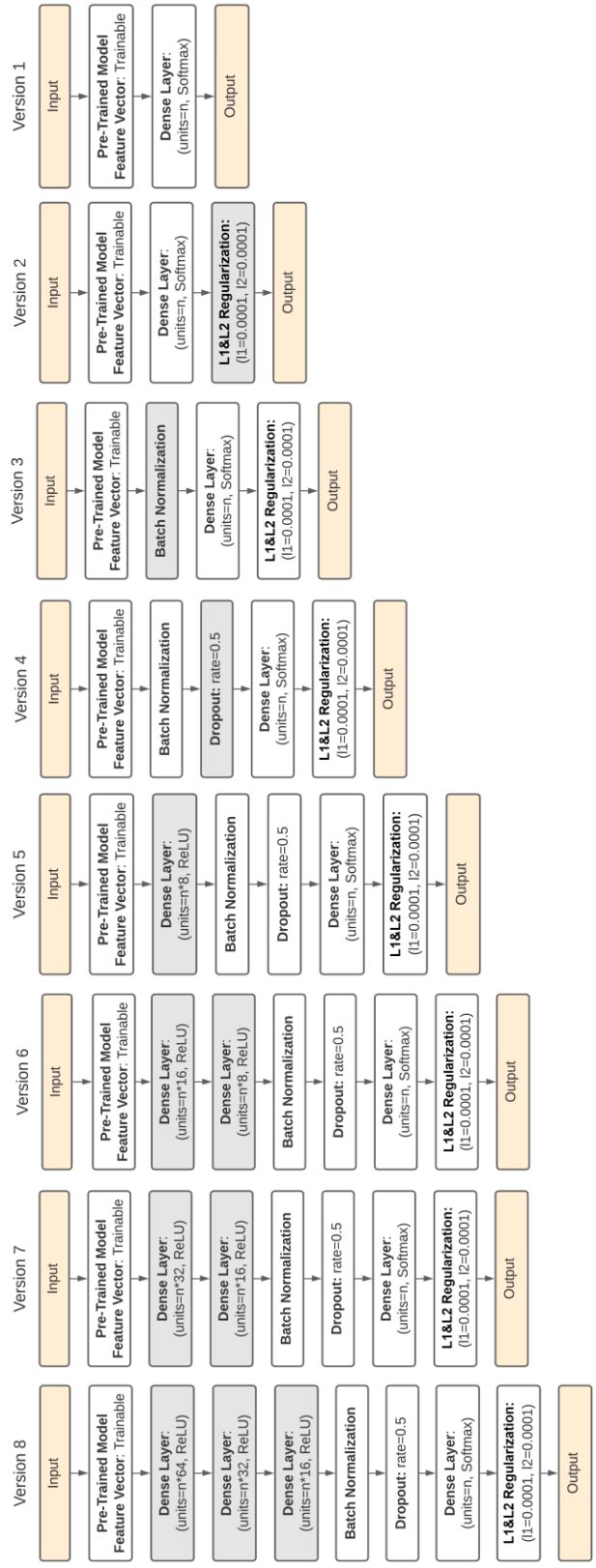


Fig. 3. Classifier versions 1-8. Here, newly added layers are colored in gray, and n denotes the number of classes.

IV. Implementation Results

This section describes the steps in building a vehicle monitoring system. The best classification architecture, which outperforms prior studies, is proposed. A detailed assessment of the classifier is provided, examining each class performance and providing some visual explanations using gradient-weighted class activation mapping. Finally, the generalizability of the model is investigated.

A. Data Description

The raw data for initial model training collected from a car sales website. The data also include pictures that are not required for the model, such as the interior of the vehicle, the wheels, and the engine. Additionally, each car has a .json file with information about the ID, make/model, body type, color, date of manufacture, and more. The images, which are all in .jpg format, also vary in size and quality. Examples of original photos of one car are shown in Fig. 4.

Fig. 4. Raw images.

B. Auto Labeler

A pre-trained YOLOv3 model on the COCO dataset with 80 classes and an input image size of 416 was used to detect image files that do not show an exterior of the car. This model was designed to search for classes such as car, bus, or truck with a preset intersection over a union threshold of 0.5 and a score threshold of 0.89. If the area of the bounding box was greater than 33 % of the image, it was considered a vehicle and cropped. A sample result is shown in Fig. 5.

Fig. 5. Cropped images around the region of interest.

C. Validation Interface

Vehicle search and image cropping operations using the YOLOv3 model did not yield perfect results. The data contained images of the car with the doors open or very close-up shots that had to be manually removed. To speed up manual work, a graphical user interface has been developed using the Tkinter Python library. Using the interface, it is possible to view images and delete poorly selected photos at the touch of a button. Examples of acceptable and unacceptable images are shown in Fig. 6.

Fig. 6. Simplified validation interface. Here, the red "Delete" button deletes the image, the green "Next" button proceeds to the next image, and the white "Undo" button returns the user back to the previous screen.

D. Image Pre-Processing

The images have been resized to 224×224 dimensions to represent the lower quality of the real-life image and to speed up model training. Each image is converted from RGB to grayscale to remove color information that is not a decisive factor in classifying a car brand. However, the pre-trained model uses 3 color channels as input, so the gray channel is repeated 3 times. Finally, the data are normalized from 0 to 1. The final dense 4-D-shaped tensor is of the following shape [batch size, height, width, color channels]. Examples of pre-processed images are shown in Fig. 7.

Fig. 7. Pre-processed images.

According to the study in [29], 200 or more images per class are preferred to achieve the best accuracy using the transfer learning strategy. Therefore, only vehicle brands with more than 400 images have been included. And excess data with more than 500 photos per class were not included to avoid difficulties with imbalanced classes. The class labels were one hot encoded. Finally, a total of 9451 images were divided by 80/20 ratio for training and testing, respectively. There were a total of 7560 train and 1891 test photos collected and 19 classes remained: Audi, BMW, Citroen, Ford, Honda, Hyundai, Kia, Lexus, Mazda, Mercedes-Benz, Mitsubishi, Nissan, Opel, Peugeot, Renault, Skoda, Subaru, Volkswagen, Volvo.

E. Data Augmentation

The data augmentation technique artificially creates modified images and is used to reduce overfitting [29]. The more data available, the better models can be developed [14]. Two different augmentation techniques were used to depict reality. The first flips the horizontal axis to make both sides of the car visible. The second method uses a random 30 % scaling to make the model more robust to small changes in object size. Figure 8 illustrates both augmentations.

Fig. 8. Zooming and horizontal flipping augmentation.

Data augmentation is used for the training dataset only, so the size of the test data remains the same (1891), but the training data increases from 7560 to 11560, about 53 % of the original data were augmented. As a result, 65.4 % of the training data are real, while 34.6 % are artificially augmented. Figure 9 shows the number of images by car brand.

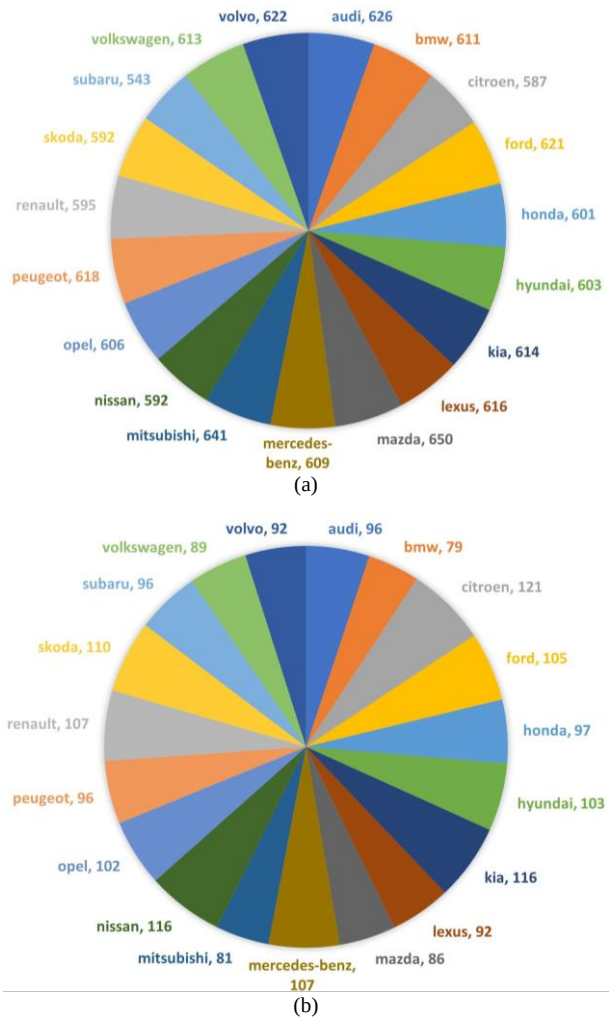


Fig. 9. Number of images in train and test datasets: (a) Number of cars in training dataset; (b) Number of cars in testing dataset.

F. Training Environment

The model is built using cloud-based Google Colab environment. The code is written in the Python programming language using the TensorFlow GPU library. The models are trained on Ubuntu 18.04.5 LTS using NVIDIA-SMI 495.44 with CUDA version 11.2. More information about the experimental setup can be found in Table I.

TABLE I. EXPERIMENTAL SETTINGS.

Attribute	Configuration Information
Operating system (OS)	Linux 5.4.144 with Ubuntu 18.04.5 LTS
CPU	Intel(R) Xeon(R) CPU @ 2.30 GHz
GPU	NVIDIA-SMI 495.44
Driver version	460.32.03
CuDNN/CUDA	8.0.5/11.2
Python version	3.7.12
Framework	Tensorflow 2.7.0

Furthermore, the initially established hyperparameters used to regulate learning processes are presented in Table II.

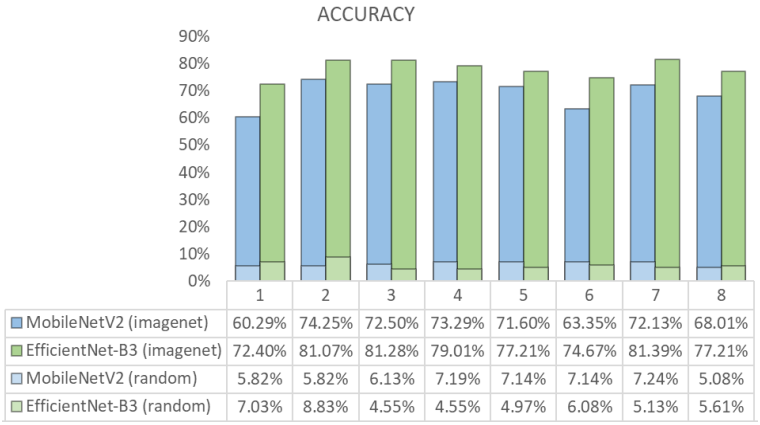
TABLE II. EXPERIMENTAL SETTINGS.

Hyperparameter	Value
Number of epochs	200
Early stopping epochs	15
Batch size	16
Optimizer	Adam
Learning rate	0.0001

The maximum number of training iterations, called "epochs" is 200. Since too many or too few can lead to overfitting or underfitting, a monitoring system was introduced with an early stop feature such that if the validation loss did not improve in the last 15 epochs, the training would be terminated. The categorical cross-entropy loss function was used for model training as an objective function to be minimized. The test loss measures the distance between the predicted probability distribution of belonging to each class and the target values. To avoid memory overflow, the batch size is set to 16 to feed the model with only a fraction of the training data at a time. Adam, an optimizer, with a learning rate of 0.0001 was selected.

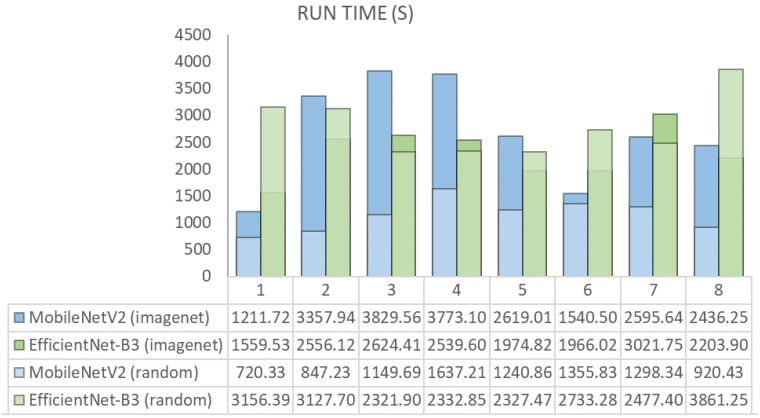
G. Testing Classifiers Versions Using Pre-Trained Models

Performing classification using the pre-trained models MobileNetV2 and EfficientNet, eight classifiers were tested. All the classifier structures are defined in Section III. Model architectures trained with random weights were also evaluated. However, due to small training data, the model with random weights did not learn the influential features and achieved the highest accuracy of 8.83 % (see Fig. 10). This demonstrates the benefits of employing pre-trained weights, which have a 12.5 times greater accuracy on average than randomly generated weights. The accuracy of the test dataset, which estimates the number of times the highest probability prediction matches the ground truth labels, is shown in Fig. 10 and the model training duration is shown in Fig. 11.



	1	2	3	4	5	6	7	8
MobileNetV2 (imagenet)	60.29%	74.25%	72.50%	73.29%	71.60%	63.35%	72.13%	68.01%
EfficientNet-B3 (imagenet)	72.40%	81.07%	81.28%	79.01%	77.21%	74.67%	81.39%	77.21%
MobileNetV2 (random)	5.82%	5.82%	6.13%	7.19%	7.14%	7.14%	7.24%	5.08%
EfficientNet-B3 (random)	7.03%	8.83%	4.55%	4.55%	4.97%	6.08%	5.13%	5.61%

Fig. 10. Comparison of the accuracy of different classifier models and versions.



	1	2	3	4	5	6	7	8
MobileNetV2 (imagenet)	1211.72	3357.94	3829.56	3773.10	2619.01	1540.50	2595.64	2436.25
EfficientNet-B3 (imagenet)	1559.53	2556.12	2624.41	2539.60	1974.82	1966.02	3021.75	2203.90
MobileNetV2 (random)	720.33	847.23	1149.69	1637.21	1240.86	1355.83	1298.34	920.43
EfficientNet-B3 (random)	3156.39	3127.70	2321.90	2332.85	2327.47	2733.28	2477.40	3861.25

Fig. 11. Run-time comparison of different classifier models and versions.

The EfficientNet-B3 feature vector and the 7th classifier had the best accuracy results of 81.39 %, although the 3rd classifier was only 0.11 % less accurate. The 7th version took 50 minutes to train, while the 3rd took 44 minutes. Although both versions are usable, the 7th has been selected. The summary of the model with the 7th classifier is shown in Fig. 12.

The layer names, type, output shape, and number of weight parameters are all listed in the model summary (see Fig. 12). At the bottom of the table is the total number of trainable and non-trainable model parameters. The "None" output value refers to the batch size. The Keras layer in this case is an invoked object for loading a stored model EfficientNet-B3 with a trainable parameter equal to "True".

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
keras_layer (KerasLayer)     (None, 1536)              12930622

dense (Dense)                (None, 608)               934496

dense_1 (Dense)              (None, 304)               185136

batch_normalization (BatchN  (None, 304)               1216
ormalization)

dropout (Dropout)            (None, 304)               0

dense_2 (Dense)              (None, 19)                5795

=================================================================
Total params: 14,057,265
Trainable params: 13,947,441
Non-trainable params: 109,824
_________________________________________________________________
```

Fig. 12. Summary of the best sequential classification model (version no. 7).

H. Training Performance of the Best Classifier

The training performance of the EfficientNet-B3 using the 7th classifier version is shown in Fig. 13.

Here, the blue line indicates the training, and the orange indicates the validation curve. The accuracy plot shows that the model gains high accuracy in the early epochs because the pre-trained weights are reused. To avoid overfitting and generalization errors, the model was trained for less than 40 epochs until the loss of validation stopped improving. However, because there is a gap between the training and validation curves, it indicates a slight overfitting, so increasing the dataset may improve the results. Additionally, the shape of the training loss curve suggests that a good learning pace was chosen.

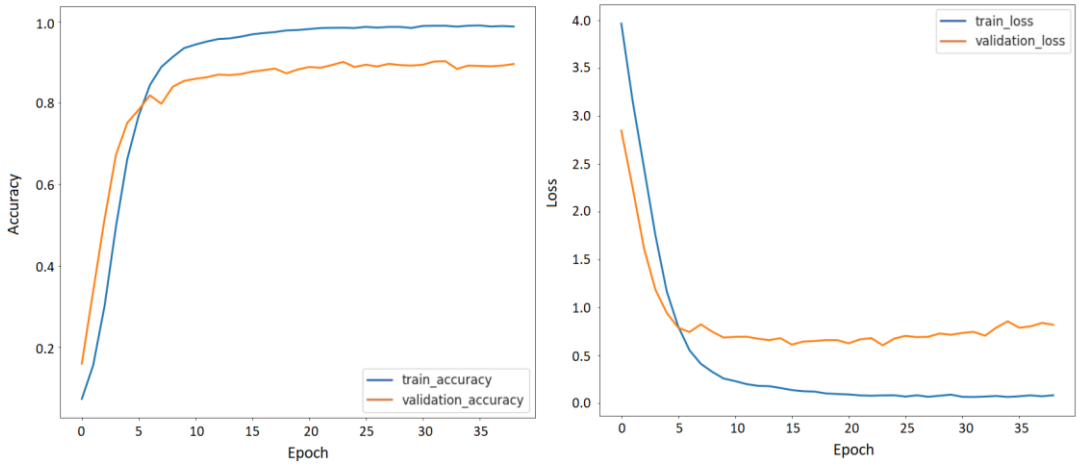


Fig. 13. Performance of the classification model training. Model performance history.

At predicted class probability thresholds ranging from 0 to 1, the receiver operating characteristic curve (ROC) displays the true positive rate on the y-axis versus the false positive rate on the x-axis. The true positive rate indicates the probability that a defined car label will be predicted correctly, and the false positive rate shows how often the prediction of a defined car is incorrect. It is usually plotted for binary classifier; therefore, the one-vs-all approach is employed for multi-class analysis. The ROC curves and the area under the curve are given in Fig. 14. Looking at the ROC curves, the model is close to a perfect classifier, as the curve is above the diagonal line.

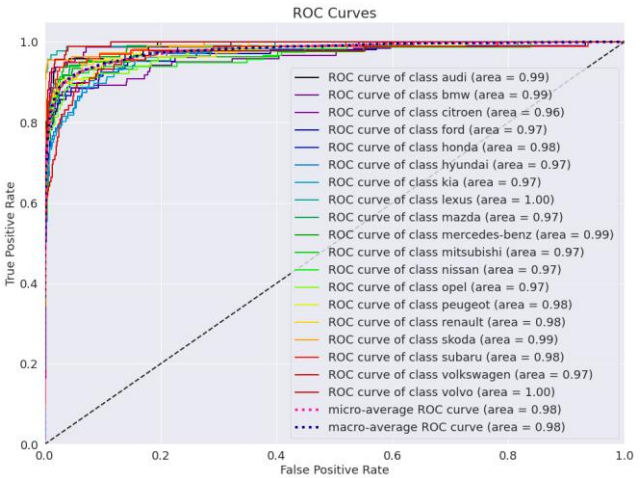


Fig. 14. Receiver operating characteristic curves.

A contrast between real and predicted values is shown in the confusion matrix (see Fig. 15).

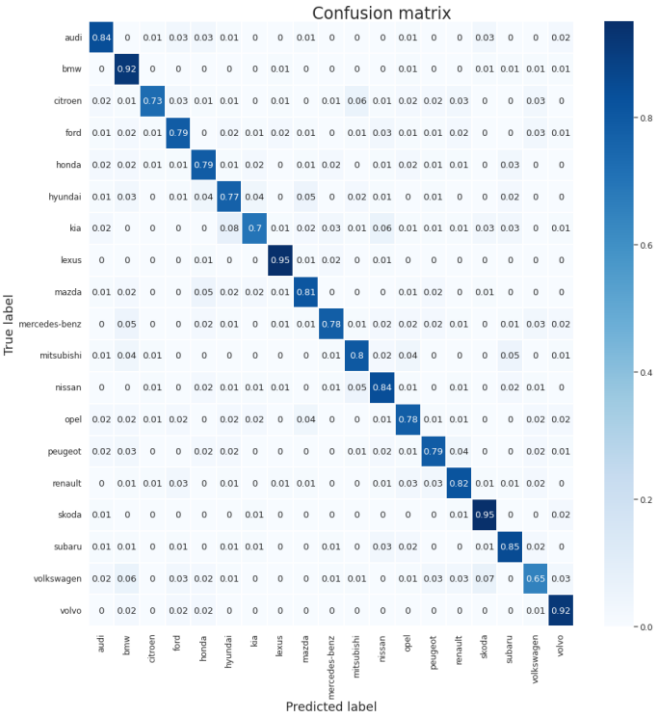


Fig. 15. Normalized confusion matrix.

Using the confusion matrix, the most confused classes could be identified and their interrelationships explored.

However, the model predictions are scattered, and the diagonal reveals the strongest link between the predictable and true classes.

I. Classifier Visual Explanations Using Grad-CAM

Deep learning models are known for their excellent accuracy, yet their complexity makes them hard to interpret. Gradient-weighted class activation mapping (Grad-CAM) is an approach to better understanding how the CNN model works. Grad-CAM descends to the final convolutional layer and uses gradient information to create a rough localization map that identifies key discriminating areas in the image for class prediction. No fully connected layers are used. The class activation mapping is then upscaled to the size of the input image so that a heat map could be presented [30].

The most important properties that determine the classifier decision are marked in red on the heat map (see Fig. 16). In the images, where the car is visible from the front, the model focuses on the brand symbol, the front grille, and a little bit on the headlights. Meanwhile, looking at the cars from the side, the model distinguishes the most discriminative regions in the middle and upper parts of the car (body, hood, and windows). To classify the car from the rear view, the model searches for the brand symbol around the license plate number. The angle of view is very important for good performance of the model. If the car is seen from above, the focus is on the body. And when the car is at a 45-degree angle, the discriminating region includes a side lamp or other car shapes.

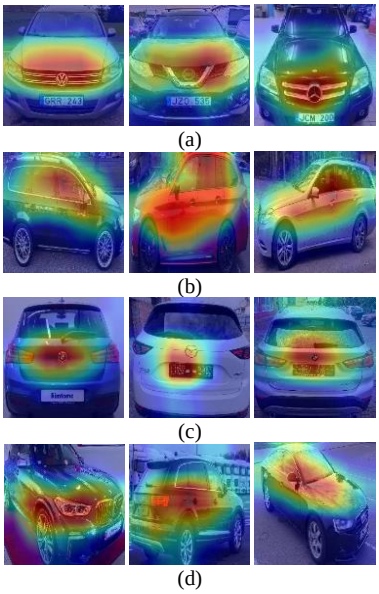


Fig. 16. Grad-CAM from different car views: (a) the front, (b) the side, (c) the back, and (d) at a 45-degree angle.

J. Generalization Testing

It is critical to check how the model works with real-world data [31]. Even if the accuracy of a test dataset is high, this does not guarantee that the model will work well in every situation. There may be a generalization gap if the model receives previously unseen data, such as different backgrounds or different weather conditions. Three different scenarios are used to verify the reliability of the model.

To start, 168 photos from the test dataset are randomly selected so that the vehicles are only seen from the side when the car brand symbol is not visible. This results in 17 classes with a 76.19 % accuracy (see Table III). As a reminder, the accuracy of the test on all data is 81.39 %, which means that the side of the car is about 5 % less predictable.

The second scenario combines both training and testing sets from the Stanford cars dataset. Because the data contained brands that our model cannot recognize, it was filtered out. That left 10 classes, the majority of which were formed of Audi and BMW vehicles. A total of 4119 pictures were acquired and an accuracy of 66.86 % was obtained.

In the third scenario, 12 photos are taken from a video recorded in the evening with artificial lighting. Six vehicle brand classes with two images each were identified with an accuracy of 75.0 %. However, due to the tiny dataset, it is advised to use more annotated video frames for validation.

TABLE III. COMPARISON OF CLASSIFICATION PERFORMANCES.

	Side Car Images	**Stanford Cars Dataset**	**Video Frames**
Test Accuracy	76.19 %	66.86 %	75.0 %
Test Loss	1.472	2.347	0.928
Total Images	168	4119	12
Number of Classes	17	10	6

Generalization testing reveals that additional datasets have a mean accuracy of 72.7 %, resulting in 8.7 % poorer estimation than the initial test dataset (81.39 %). In general, the model can be generalized, but the accuracy will not be as good as for the same training distribution. As a result, integrating diverse data distributions in the training phase can help the model adapt to a broader range of scenarios.

V. DISCUSSION

In this paper, the collection of country-based datasets and a strategy for vehicle localization and brand classification are proposed. The data represent the most frequent cars on the Lithuanian market and were balanced to avoid biases. The data preparation stage was automated using the YOLOv3 car detection model; however, the manual image validation stage could be eliminated by optimizing the model parameters and making it more conservative. Analyzing the classifiers, it was discovered that EfficientNetV2 is 7.14 % more accurate than MobileNetV2. The 7th version of the classifier, consisting of three dense layers, batch normalization, dropout, and L1 and L2 regularization, improved the most in accuracy. Compared to the baseline classifier, the EfficientNetV2 7th classifier improved its accuracy by 9 % (from 72.4 % to 81.4 %). This was accomplished by classifying 19 automobile manufacturer classes using a batch size of 16 and 38 epochs. Furthermore, the architecture created outperforms the maximum accuracy of the fully trained GoogLeNet architecture of 80 % [10], although the performance cannot be directly compared due to different datasets and specific configurations.

After generalization tests using the EfficientNetV2 7th classifier, the average accuracy of three different datasets was 72.7 %. Compared to the accuracy of the original dataset testing sample, the accuracy dropped by 8.7 %. The reduction in accuracy is not significant, and some

fluctuation is expected. As observed from the results of the Grad-CAM method, the model is trained to recognize the brand of the car by the location of the logo, and if it is not found, the car body, hood, or windows become an essential predictor. On the other hand, mixing various data into a training dataset can help strengthen model generalization abilities.

The following are suggestions for future improvements:

– More innovative augmentation techniques might be proposed to increase generalization performance in diverse weather and lighting settings. For example, artificially manipulating photos to make them appear as though they were taken at night, with a darker background and light reflections on the car. In addition, it would be useful to add more data from different sources, representing different nature of images, diverse quality, viewpoints, and so on. As background clutter reduces model performance [12], removing it improves vehicle classification results [1]. By changing the eliminated background with various representations of road pictures, this technique might be utilized in augmentation.

– As one study was able to classify the position of a car with 100 % accuracy [16], it could be re-used effectively to classify cars from different angles. The suggestion would be to first classify the visible position of the car and then, depending on the results, classify the features of the car using several trained models. As a result, the learnt parameters of the model would be unique for each given viewpoint, although it requires labeled data on the viewpoint of the vehicle.

– The task could also be expanded to include other vehicle attributes, such as vehicle type, make, model, or color, so that the vehicle can be precisely identified. It would also be beneficial to test the model using video data.

VI. CONCLUSIONS

In this paper, a vehicle detection and classification system was suggested and implemented, which was found to be able to classify common local vehicle brands regardless of the viewable angle. This is especially relevant as smart city systems and the use of security and traffic monitoring cameras become more widespread. To represent the country's most popular car manufacturers, static images from the Lithuanian car sales website were used. However, this step of data collection has also become a barrier in the system, as it requires manual validation, which should be replaced by an automated and more efficient data labeling solution. Effective deep learning architectures have been investigated to make the model easier to use in real time and to incorporate new car classes as production grows. The proposed EfficientNetV2 architecture adjustment improves the performance of the original classifier by 9 % and achieves an acceptable classification score of 81.4 %. However, to adjust the model to real-world conditions, data from the environment in which it will be used should be incorporated into the training. The findings imply that the trained model can be used for urban vehicle monitoring, and various improvements are proposed for future research.

CONFLICTS OF INTEREST

The authors declare that they have no conflicts of interest.

REFERENCES

[1] M. A. Manzoor, Y. Morgan, and A. Bais, "Real-time vehicle make and model recognition system", *Machine Learning and Knowledge Extraction*, vol. 1, no. 2, pp. 611–629, 2019. DOI: 10.3390/make1020036.

[2] S. Benslimane, S. Tamayo, and A. de La Fortelle, "Classifying logistic vehicles in cities using Deep learning", in *Proc. of 15th World Conference on Transport Research*, 2019, vol. 33. arXiv: 1906.11895.

[3] M. A. Butt *et al.*, "Convolutional neural network based vehicle classification in adverse illuminous conditions for intelligent transportation systems", *Complexity*, vol. 2021, art. ID 6644861, 2021. DOI: 10.1155/2021/6644861.

[4] N. Yaras, "Vehicle type classification with deep learning", M.S. thesis, İzmir Institute of Technology, İzmir, 2020.

[5] S. Naseer, S. M. A. Shah, S. Aziz, M. U. Khan, and K. Iqtidar, "Vehicle make and model recognition using deep transfer learning and support vector machines", in *Proc. of 2020 IEEE 23rd International Multitopic Conference (INMIC)*, 2020, pp. 1–6. DOI: 10.1109/INMIC50486.2020.9318063.

[6] A. B. Khalifa and H. Frigui, "A dataset for vehicle make and model recognition", in *Proc. of 3rd Workshop Fine-Grained Vis. Categorization (FGVC)*, 2015, pp. 1–2.

[7] S. Baghdadi and N. Aboutabit, "View-independent vehicle category classification system", *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 7, pp. 756–771, 2021. DOI: 10.14569/IJACSA.2021.0120786.

[8] B. Satar and A. E. Dirik, "Deep learning based vehicle make-model classification", in *Artificial Neural Networks and Machine Learning – ICANN 2018. ICANN 2018. Lecture Notes in Computer Science*, vol. 11141. Springer, Cham, 2018. DOI: 10.1007/978-3-030-01424-7_53.

[9] D. Liu and Y. Wang, "Monza: Image classification of vehicle make and model using convolutional neural networks and transfer learning", 2016. [Online]. Available: http://cs231n.stanford.edu/reports/2015/pdfs/lediurfinal.pdf

[10] "U.S. new car model market launches 2021 | Statista". [Online]. Available: https://www.statista.com/statistics/200092/total-number-of-car-models-on-the-us-market-since-1990/

[11] M. M. Hasan, Z. Wang, M. A. I. Hussain, and K. Fatima, "Bangladeshi native vehicle classification based on transfer learning with deep convolutional neural network", *Sensors*, vol. 21, no. 22, p. 7545, 2021. DOI: 10.3390/s21227545.

[12] S. Hashemi, H. Emami, and A. B. Sangar, "A new comparison framework to survey neural networks-based vehicle detection and classification approaches", *International Journal of Communication Systems*, vol. 34, no. 14, p. e4928, Sep. 2021. DOI: 10.1002/dac.4928.

[13] M. Guerrieri and G. Parla, "Deep learning and YOLOv3 systems for automatic traffic data measurement by moving car observer technique", *Infrastructures*, vol. 6, no. 9, p. 134, 2021. DOI: 10.3390/infrastructures6090134.

[14] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning", *Journal of Big Data*, vol. 6, no. 1, Dec. 2019. DOI: 10.1186/s40537-019-0197-0.

[15] C. Desai, "Image classification using transfer learning and deep learning", *International Journal of Engineering and Computer Science*, vol. 10, no. 9, pp. 25394–25398, Sep. 2021. DOI: 10.18535/ijecs/v10i9.4622.

[16] S. Baghdadi and N. Aboutabit, "Transfer learning for classifying front and rear views of vehicles", *Journal of Physics: Conference Series*, vol. 1743, no. 1, p. 012007, 2021. DOI: 10.1088/1742-6596/1743/1/012007.

[17] A. Gupta and M. Gupta, "Transfer learning for small and different datasets: Fine-tuning a pre-trained model affects performance", *Journal of Emerging Investigators*, vol. 3, 2020.

[18] V. Taormina, D. Cascio, L. Abbene, and G. Raso, "Performance of fine-tuning convolutional neural networks for HEp-2 image classification", *Applied Sciences*, vol. 10, no. 19, p. 6940, 2020. DOI: 10.3390/app10196940.

[19] Y. Gao and H. J. Lee, "Local tiled deep networks for recognition of vehicle make and model", *Sensors*, vol. 16, no. 2, p. 226, 2016. DOI: 10.3390/s16020226.

[20] H. J. Lee, I. Ullah, W. Wan, Y. Gao, and Z. Fang, "Real-time vehicle make and model recognition with the residual SqueezeNet

architecture", *Sensors*, vol. 19, no. 5, p. 982, 2019. DOI: 10.3390/s19050982.
[21] F. Tafazzoli, "Vehicle make and model recognition for intelligent transportation monitoring and surveillance", 2017. DOI: 10.18297/etd/2630.
[22] F. Xu, K. Han, Y. Ruan, and Y. Mao, "Car classification using neural networks", pp. 1–6, 2000.
[23] H. Wang, Q. Xue, T. Cui, Y. Li, and H. Zeng, "Cold start problem of vehicle model recognition under cross-scenario based on transfer learning", *Computers, Materials and Continua*, vol. 63, no. 1, pp. 337–351, 2020. DOI: 10.32604/cmc.2020.07290.
[24] F. Pereira dos Santos and M. Antonelli Ponti, "Features transfer learning for image and video recognition tasks", in *Proc. of Workshop de Teses e Dissertações - Conference on Graphics, Patterns and Images (SIBGRAPI)*, 2020, pp. 29–35. DOI: 10.5753/sibgrapi.est.2020.12980.
[25] Z. Qin, Z. Zhang, X. Chen, C. Wang, and Y. Peng, "Fd-Mobilenet: Improved Mobilenet with a fast downsampling strategy", in *Proc. of 2018 25th IEEE International Conference on Image Processing*, 2018, pp. 1363–1367. DOI: 10.1109/ICIP.2018.8451355.
[26] M. Tan and K. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks", in *Proc. of International Conference on Machine Learning*, 2019. DOI: 10.48550/arXiv.1905.11946.
[27] M. Tan and Q. V. Le, "EfficientNetV2: Smaller Models and faster training", in *Proc. of International Conference on Machine Learning*, 2021. DOI: 10.48550/arXiv.2104.00298.
[28] T. Ridnik, E. Ben-Baruch, A. Noy, and L. Zelnik-Manor, "ImageNet-21K pretraining for the masses", 2021. DOI: 10.48550/arXiv.2104.10972.
[29] Y. Kang, N. Cho, J. Yoon, S. Park, and J. Kim, "Transfer learning of a deep Learning model for exploring tourists' urban image using geotagged photos", *ISPRS International Journal of Geo-Information*, vol. 10, no. 3, p. 137, 2021. DOI: 10.3390/ijgi10030137.
[30] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization", *International Journal of Computer Vision*, vol. 128, pp. 336–359, 2020. DOI: 10.1007/s11263-019-01228-7.
[31] A. Harras, A. Tsuji, S. Karungaru, and K. Terada, "Enhanced vehicle classification using transfer learning and a novel duplication-based data augmentation technique", *International Journal of Innovative Computing, Information and Control*, vol. 17, no. 6, pp. 2201–2216, 2021. DOI: 10.24507/ijicic.17.06.2201.