Car make and model recognition using convolutional neural network: fine-tune AlexNet architecture

Md. Asifur Rahman Khan[1], Raju Talukder[1], Md. Anwar Hossen[1,3], Nusrat Jahan[2,4]
[1]Department of Software Engineering, Faculty of Science and Information Technology, Daffodil International University, Dhaka, Bangladesh
[2]Department of Information Technology and Management, Faculty of Science and Information Technology, Daffodil International University, Dhaka, Bangladesh
[3]Department of Computer Science, Iowa State University, Ames, United State
[4]School of Computer and Communication Engineering, Universiti Malaysia Perlis, Arau, Malaysia

Article Info

Article history:

Received Aug 30, 2022
Revised Oct 26, 2023
Accepted Nov 12, 2023

Keywords:

AlexNet
Car model recognition
Convolutional neural network
Deep learning
Fine tuned-AlexNet

ABSTRACT

Artificial intelligence (AI) has significantly contributed to car make and model recognition in this current era of intelligent technology. By using AI, it is much easier to identify car models from any picture or video. This paper introduces a new model by fine-tuning the AlexNet architecture to determine the car model from images. First of all, our car image dataset has been created. Some of these images were taken by us, and others were taken from the website of the car connection. Then we cleaned all the unwanted images for better performance. Our dataset has ten classes containing 5,000 car images split into train and test data. After that, we augmented our data with random flip, rotation, and zoom to reduce overfitting. Finally, we used a pre-trained convolutional neural network (CNN) model AlexNet architecture. We fine-tuned AlexNet (FT-AlexNet) by adding three extra layers for better classification and compared it with the original AlexNet. To measure the performance of these models, accuracy, precision, recall, and F1-score were used. The results show that fine-tune AlexNet architecture outperforms the original AlexNet architecture. The results prove that recognition accuracy has increased due to our improvement approach.

Corresponding Author:

Md. Asifur Rahman Khan
Department Software Engineering, Faculty of Science and Information Technology
Daffodil International University
Dhanmondi-19, Dhaka, Bangladesh
Email: asifur35-2259@diu.edu.bd

1. INTRODUCTION

In recent years, car make and model recognition (CMMR) holds the most exciting research topic in the computer vision field. CMMR is a subset of vehicle make and model recognition (VMMR). VMMR recreates an essential position in intelligent transportation systems (ITS) and automated vehicular surveillance (AVS), such as vehicle management, illegal hit and run, self-driving, auto-parking, automated tolling systems, and traffic surveillance. The main challenge in decoding this trouble lies in the significant variability of vehicle models, which significantly increases the variety of images [1]. Many features can be extracted to recognize a car from an image, such as license plate, headlights, tires size. However, if these car features are not visible clearly, it becomes challenging to identify due to low light. Consequently, these have some shortcomings also. Even in limited lighting conditions, the whole body of the car can be seen quite well. So, it might be more beneficial to work with the car's entire body than with different features. As a result, recognition accuracy should be promising too. Cars have many unique properties, which provide many challenging and several issues in the make and model recognition research field of vehicles as different models of vehicles from the same company are often of the same size, shape, and design [2]. Complexities may also occur due to the different positions of the vehicles in the image.

So far, many models have been introduced for car make and model detection. However, they are often traditional methods like feature extraction and machine learning algorithms. Moreover, these methods also have some time-consuming, low accuracy, robustness, and convenience issues [3]. With the development of deep learning and computer vision technologies like artificial neural network (ANN) and convolutional neural network (CNN), the recognition of vehicle types has been extensively used. Therefore, need to capture the images of the cars and feed them to the computer by any media, and the computer classifies the image using computer vision algorithms. However, most of the technologies in practical life use traditional algorithms where computer vision and deep learning algorithms are much more efficient and timesaving. In continuation of this, we build our dataset. Deep learning is beneficial over these methods, since it does not require handcrafted features. Instead, it detects the optimal feature set on its own for a given issue or problem for classification. Therefore, the objective is to expand the field of CMMR and explore more using deep learning methods through AlexNet pre-trained architecture and fine-tuning it to obtain a better classification result than AlexNet.

Due to its possible practical benefits and scientific importance, CMMR has attracted the interest of numerous academics and engineers over the last 10–15 years. Many models have been introduced for car make and model detection till now. However, they are usually traditional methods like feature extraction and machine learning methods. Furthermore, these methods also have some shortcomings. Therefore, in this context, deep neural networks work better than these. Though there are only a few works on CMMR using deep neural networks and CNNs. However, the amount of related scientific papers has been comparatively small in recent years. This segment discusses previous work on car model recognition and image classification.

Methods used for CMMR can be divided into three generic approaches: traditional, machine learning, and neural network-based methods. Identifying feature points inside the image has recently been proposed using a variety of descriptors based on traditional methodologies [4]. Scale-invariant feature transform (SIFT) [5] connects a scale-invariant region pointer and a descriptor based on the gradient allocation in the detected regions. Then edge-based features [6], [7], speeded-up robust features (SURF) [8], [9], oriented fast and rotated brief (ORB) [10], and histogram of oriented gradient (HOG) [11] have been used to descript the car model images. Feature descriptors encode data into a sequence of numbers that can distinguish one feature from another. Based on crucial features determination and description, these algorithms provide features independent of light, scale, noise, and rotation variations and then use Euclidian distance between descriptors for coordinating images. Zhang [12] proposed a two-stage cascade classifier ensemble with a refusal option based on a pyramid HOG and Gabor features derived from frontal images of cars to support the conditions when no decision should be made once sufficient ambiguity occurs.

A variety of machine learning classifiers have also been introduced to classify objects like k-nearest neighbor (KNN) [13], support vector machines (SVM) [14], and Bayesian method [15]. However, these were not used for car model recognition to our knowledge. Besides, some works used combinations of two or more features to gain better results, such as combining wavelet and contourlet features [16] and a combination of pyramid histogram of oriented gradients (PHOG) and Gabor features [12]. Yang *et al.* [17] used CNN feature+joint Bayesian and CNN feature+SVM to detect cars features.

At present, the application of neural networks in image classification has increased a lot. Huttunen *et al.* [18] proposed a deep neural network (DNN) that extracts features from a car image with the background and placing a bounding box around the car. To our knowledge, there are not many previous efforts on the car model recognition task using CNN. However, several works on various image recognition using CNN are much similar to car model recognition. Gao and Lee [19] combine the frame difference method with CNN, and then results come from frame differencing are the binary images applied to identify the cars. Wang *et al.* [20] proposed vehicle type classification using faster regional based CNN (Faster R-CNN). Then there are some popular pre-trained models for image classification, such as AlexNet [21], visual geometry group (VGG16) [22], ResNet [23] and GoogLeNet [24]. These models have better accuracy, robustness, and speed than the traditional and machine learning models.

By reviewing the works performed by various authors, it has been found out that there are not so many novel works on car make and model recognition. So, in this study, we introduced a novel approach for car model recognition by fine-tuning the AlexNet architecture. We curated a comprehensive dataset of car images specifically for this task. Building upon the pre-trained CNN model AlexNet, we enhanced its performance by incorporating three additional layers designed to improve classification accuracy. We then thoroughly compared our fine-tuned AlexNet (FT-AlexNet) architecture and the original AlexNet.

The paper is organized in the following sequence. Section 2 discusses the dataset, proposed methodology, and its architectural diagrams. The results of the experiments are discussed in section 3 with diagrams. Lastly, in section 4 the conclusion is discussed.

2. RESEARCH METHOD

2.1. Dataset

As the human brain, a CNN needs much pertinent information before recognizing an image [3]. Hence, in this research, we should first build the image dataset of vehicle models so that the CNN model can learn from the dataset. Despite continuous research, the heterogeneous dataset of car make and model recognition in computer vision receives insufficient attention. Half of the images used in this research are taken from the website of the car connection [25]. The other half of the images are taken by ourselves. Then the dataset is built by combining the images collected from the website and ourselves. This dataset contains photos captured by different users, devices, and several viewpoints, assuring a broad range of variations. The cars are not well placed, and some images have irrelevant backgrounds. These images are all red, green and blue (RGB) images, as shown in Figure 1 as sample images. Figures 1(a) and 1(b) show a right front view and left front view samples, whereas Figures 1(c) and 1(d) show a rear view and front view samples of a car. The resolution of the images is all different, which can help increase the model's performance. This dataset has a total of 5,000 images arranged in 10 different classes. For each category, 500 images have used for training and testing purposes.

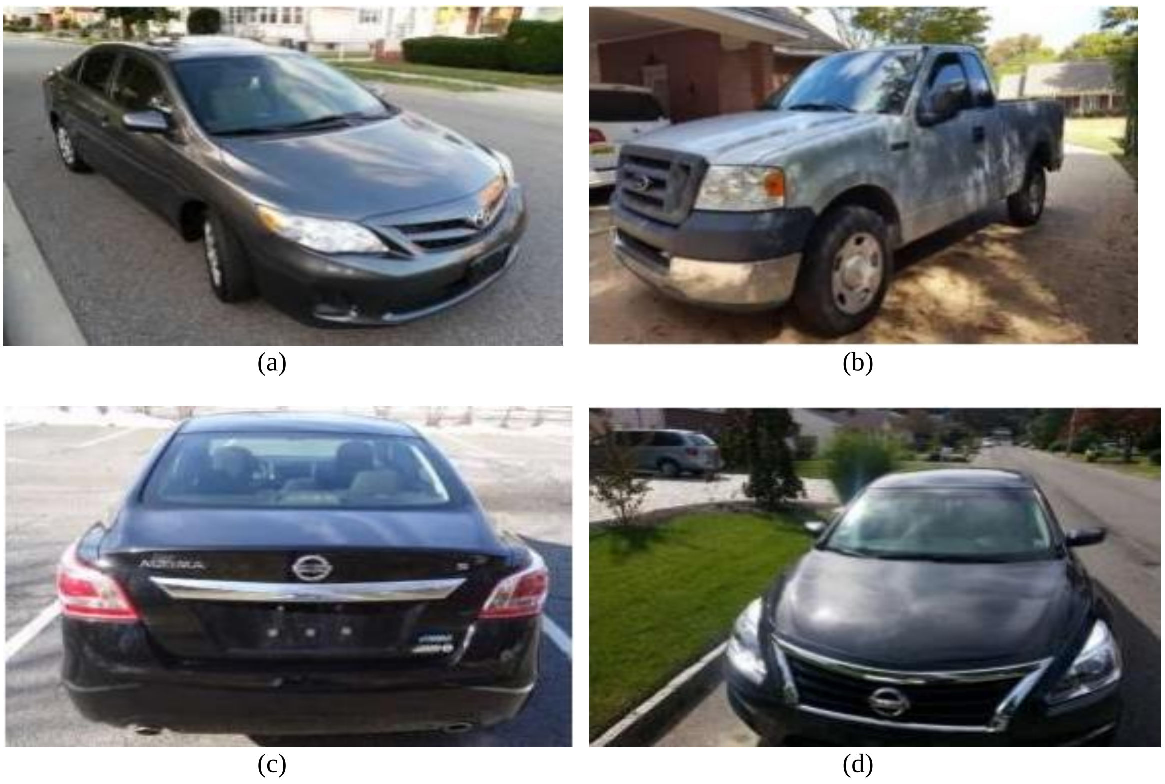

Figure 1. Four different car model images from our dataset: (a) is a right front view sample, (b) is a left front view sample, (c) is a rear-view sample, and (d) is a front view sample

One noticeable thing is that the same automobile manufacturer brand does not always correspond to the same class of car models, such as Toyota Camry and Corolla. The brand and model of all the cars in our dataset with the number of vehicles are listed in Table 1. Our dataset is much challenging because there are 2-3 different models of cars that look almost the same. Before feeding the model, all the unwanted images were removed from the dataset.

Table 1. Car model category in our dataset

Car manufacturer company	Car model	Number of images
Chevrolet	Impala	500
	Silverado	500
Dodge	Ram	500
Ford	F150	500
GMC	Sierra	500
Honda	Accord	500
	Civic	500
Nissan	Altima	500
Toyota	Camry	500
	Corolla	500
	Total	5,000

2.2. Convolutional neural network

CNN is a benchmark of deep learning. The animal visual cortex is the inspiration for CNN, which evaluates data with a grid pattern such as images [26], [27]. CNN is designed to automatically and adaptively learn the spatial ordering of characteristics from low-level to high-level patterns [28]. That is why CNN works better than all the other traditional algorithms. CNN does not need a separate feature extraction model.

2.3. AlexNet architecture

A pre-trained model is a model developed and trained by someone else to address a related issue. They usually choose a large dataset over millions of images to train these models. Then we extract these models to solve our problems. This process is called transfer learning. In 2012, Krizhevsky *et al.* [21] introduced the AlexNet architecture. It prevailed in the 2012 ImageNet large scale visual recognition challenge (ILSVRC) contest by a considerable margin. It was a revolution in artificial intelligence (AI) at that time. AlexNet architecture consists of three maxpooling layers, three fully connected layers, five convolutional layers and one softmax layer. They also used batch normalization and dropout.

The input for AlexNet's first convolutional layer is a (227×227×3) image. 96 (11×11) kernels are used in the convolution operation with no padding and four stride. As shown in Figure 2, this generates a (55×55×96) output and is passed to rectified linear unit (ReLU). We can calculate this with (1):

$$\frac{n+2p-f}{s}+1 \tag{1}$$

where n denotes the image size, p stands for padding, f for filter size, and s for stride.

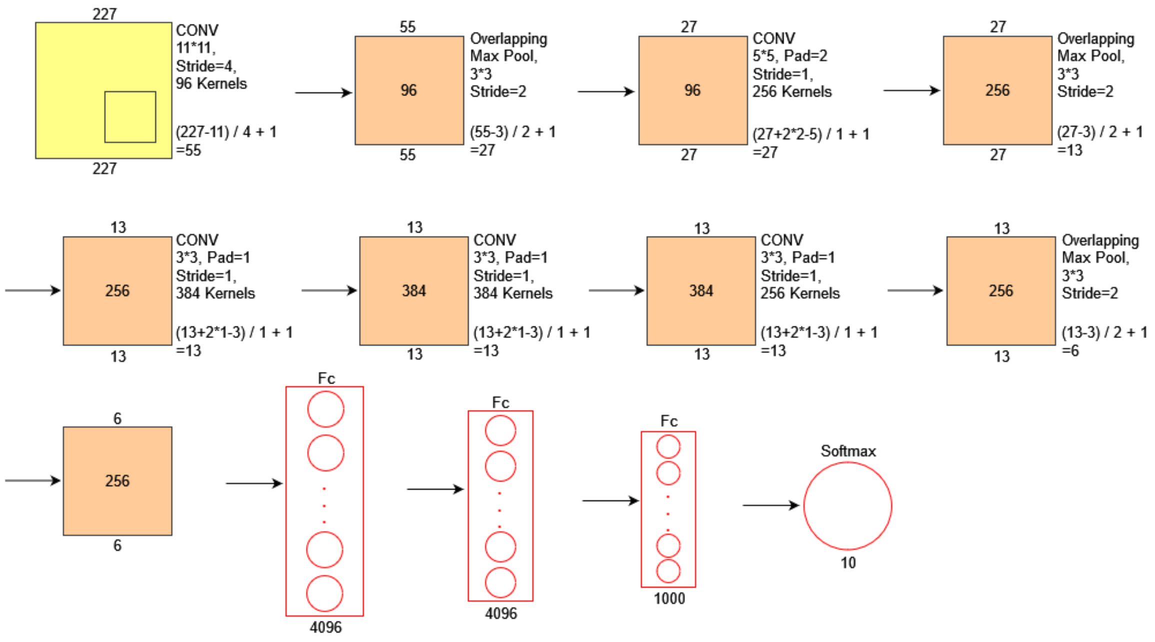

Figure 2. An illustration of AlexNet architecture

A positive or zero linear output function is called ReLU activation function. Otherwise, the values will be removed. It is used to increase the network's nonlinearity [18]. The second layer is a max-pooling layer. Max pooling determines the maximum or average value of a local feature map. It extracts low-level features such as edges and points and contributes to increase the robustness of CNN. Here it accepts (55×55×96) input from the first layer. It operates using a (3×3) kernel with no padding and a stride of two. This produces a (27×27×96) output that is passed on to the next layer. The third layer is another convolutional layer that accepts the output (27×27×96) from the previous layer. It performs a convolution operation using 256 (5×5) kernels with one stride and padding of two. This generates a (27×27×256) output and is passed to a ReLU. Then the fourth layer is a max-pooling layer that accepts (27×27×256) input from the third layer. It operates using a (3×3) kernel with no padding and two strides, similar to the 2nd layer.

This produces a (13×13×256) output that is passed on to the next layer. The fifth layer is another convolutional layer that accepts the output (13×13×256) from the previous layer. It performs a convolution operation using 384 (3×3) kernels with a stride and padding of one. This generates a (13×13×384) output and is passed to a ReLU activation function. The sixth layer is a convolutional layer which is as same as the fifth layer. It also generates the same output to a ReLU. The seventh layer is another convolutional layer that accepts the output (13×13×384) from the previous layer. It performs a convolution operation using 256 (3×3) kernels with a stride and padding of one. This generates a (13×13×256) output and is passed to a ReLU activation function. The eighth layer is a max-pooling layer that accepts (13×13×256) input from the third layer. It performs using a (3×3) kernel with no padding and two strides. This produces a (6×6×256) output that is later carried away to the fully connected layer.

After generating the output of each layer, there is a layer of batch normalization. It is used to normalize the output of the prior layer. Then the output layer in neural networks uses softmax as an activation method. It is a function of the distribution of discrete stochastic variables that can take n possible values and normalizes the distribution [29]. Additionally, some hyper-parameters have been used on a trial-and-error basis for better performance. We used Nesterov-accelerated adaptive moment estimation (Nadam), a better version of adaptive moment estimation (Adam) with a learning rate 0.001.

2.4. Fine-tune AlexNet

Fine-tuning is a way of utilizing transfer learning. Transfer learning uses the knowledge gained from one completed task to solve a different but related assignment. In this regard, we have fine-tuned the AlexNet model to give better results for our car image dataset. The low layers in the CNN are utilized to acquire some edge and corner characteristics of input images, while the tops are for the combination of these features. The top neurons represent the image's general characteristics [3], [30]. Figure 3 shows the differences between the original AlexNet and FT-AlexNet. Figure 3(a) is the original AlexNet model structure, and Figure 3(b) is the FT-AlexNet model structure. As indicated in Figure 3(b), we implemented two convolutional layers with ReLU and a max-pooling layer.

Previously AlexNet is discussed in detail in section 3.3. The improved section of FT-AlexNet will be discussed in detail here. As mentioned in section 3.3, we have added another extra convolutional layer after the third convolutional layer, as shown in Figure 4. This additional layer accepts the output (27×27×256) from the third layer. It executes a convolution operation using 256 (3×3) kernels with one stride and one padding. This produces an output of (27×27×256), which is then given to an additional ReLU activation function. Then a max-pooling layer accepts (27×27×256) input from the extra convolutional layer, the fourth layer. Then the model structure is the same up to the end of the fifth layer as discussed in section 3.3. After the fifth layer, we have generated an additional max-pooling layer and an additional convolutional layer. The output (13×13×384) from the fifth layer is accepted by this additional max-pooling layer. It runs on a (3×3) kernel with a stride of one and zero padding. This generates an (11×11×384) output that is passed on to the next layer. Then the other additional convolutional layer accepts the output (11×11×384) from the previous additional max-pooling layer. It performs a convolution operation utilizing 384 (5×5) kernels with one stride and one padding. This produces a (9×9×384) output, which is then given to an additional ReLU activation function, as depicted in Figure 4. Then the sixth layer is a convolutional layer which is as same as the previous additional convolutional layer. It also generates the same output to a ReLU. The seventh layer is another convolutional layer that accepts the output (9×9×384) from the previous layer. It uses 256 (3×3) kernels, a stride, and one padding to execute a convolution operation. It produces a (9×9×256) and is given to a ReLU activation function. The seventh layer's (9×9×256) input is accepted by the eighth layer, which is a max-pooling layer. It performs with a (3×3) kernel, two strides, and no padding. This gives an output of (4×4×256) that is passed on to the fully connected layer. The remainder is identical to section 3.3.

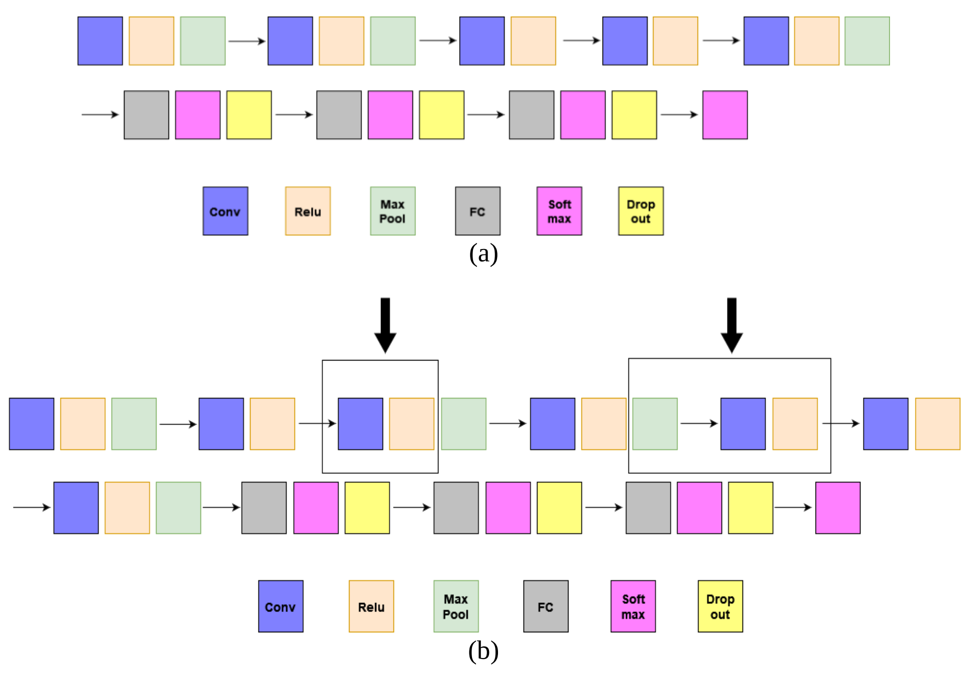

Figure 3. Differences between AlexNet and FT-AlexNet model structure (a) AlexNet model structure and (b) FT-AlexNet model structure

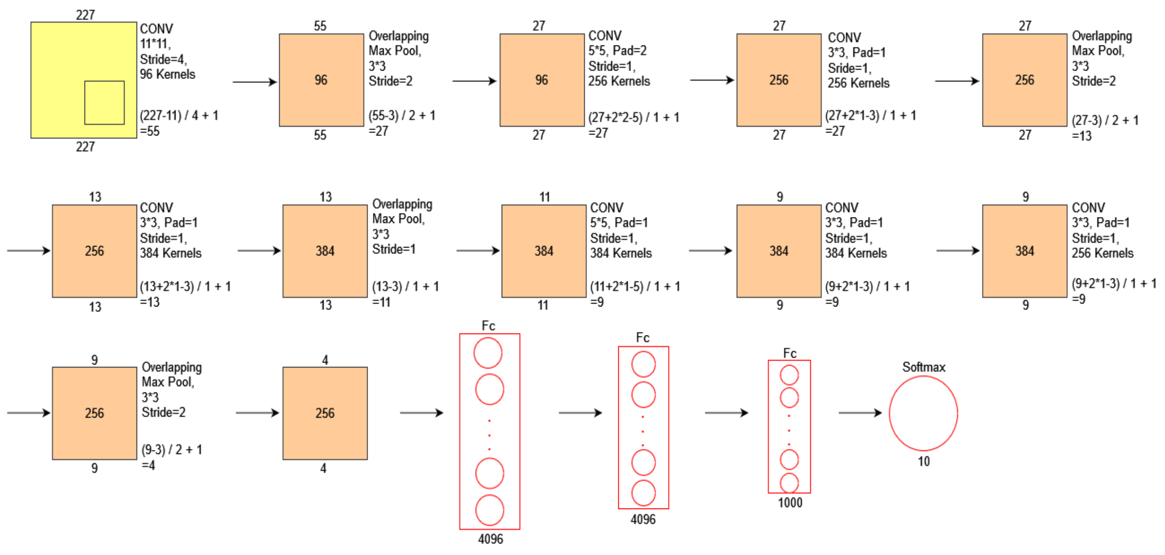

Figure 4. An illustration of improved FT-AlexNet architecture

3. RESULTS AND DISCUSSION

AlexNet and FT-AlexNet models have been tested on our dataset. The data augmentation technique and dropout layer are used to reduce overfitting. Data augmentation helps to improve the performance and outcomes of models by creating new and various data to train the dataset. We used random flip, random rotation, and random zoom to create new different image examples. Furthermore, dropout is a regularization technique that aims to decrease the complexity and ambiguity of any model to prevent over-fitting.

3.1. Performance evaluation metrics

The next stage is to assess the model's efficacy using some metrics after data preparation, applying these two models, and receiving the output. We used precision and recall, F1 score, and accuracy in evaluating our models. We also distinguished the outcomes of AlexNet with FT-AlexNet.

Precision and recall: precision specifies that out of total positive predictions, what is the percentage of the actual accurate result. To calculate precision, we can use (2).

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

The rate of precisely anticipated positive observations divided by the total number of observations in the class is known as recall. We can use (3) to determine recall.

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

F1-score: the F1-score is calculated as the weighted average of precision and recall. As a result, both false positive (FP) and false negative (FN) are considered in this score. The F1-score is frequently more helpful than the accuracy. This is defined as (4).

$$F1 = \frac{2\times(Percision\times Recall)}{Percision+Recall} \tag{4}$$

Accuracy: the percentage of accurately anticipated findings to all findings is called accuracy. It is the most basic and widely used performance metric. To calculate accuracy as (5).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100 \tag{5}$$

3.2. Discussion

The performance of AlexNet and the enhanced FT-AlexNet models are compared using the accuracy, precision, recall, and F1-score. In our dataset, 80% of the photos were used for training, while the remaining 20% were used for testing. With these two models, the first comparison is performed using epoch=100, batchsize=32, and optimizer=Nadam for each model. And the model is trained with a 0.001 learning rate. The goal of this comparison is to see if the adjustments made to AlexNet's high-level and mid-level layers have any impact on accuracy and loss. So, in that case, the results show some significant performance changes. With the AlexNet model, Figure 5 demonstrates 73.6% accuracy on the training phase and 64.08% on testing. Then, using the FT-AlexNet model, 96.56% accuracy is achieved in training, and 80.80% on the testing, as shown in Figure 6. Data show that the accuracy rate increases by 16% to 17% for the testing set after fine-tuning when using our improved network. We also measured the loss by these two models. For better understanding, model accuracy and model loss of AlexNet and FT-AlexNet are listed in Table 2. The comparison shows that our FT-AlexNet has performed better than the original AlexNet architecture.

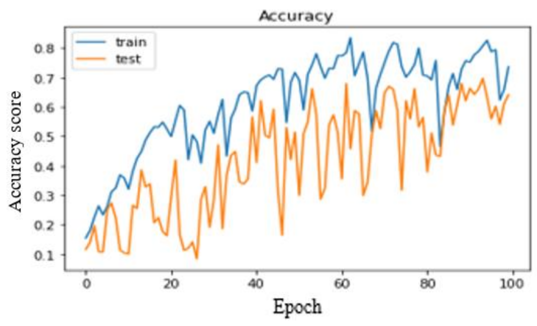

Figure 5. Model accuracy in AlexNet

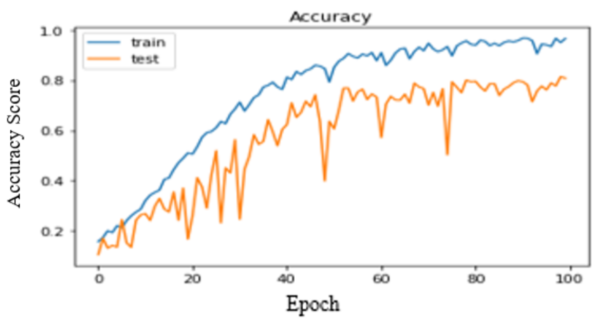

Figure 6. Model accuracy in FT-AlexNet

Table 2. Model accuracy and loss of AlexNet and FT-AlexNet

	AlexNet	FT-AlexNet
Training accuracy	73.6%	96.56%
Testing accuracy	64.08%	80.8%
Training loss	0.76	0.1002
Testing loss	1.26	0.92

For easier comparison, Table 3 provides additional performance measures for the AlexNet and FT-AlexNet models, including precision, recall, and F1-score. From Table 3, It can be observed that, FT-AlexNet classifies more accurately than AlexNet for each car model category. Figures 7 and 8 is the confusion matrix of AlexNet and FT-AleNet models, showing the difference between the original truth value and the model's predicted value. The performance difference between these two models is clearly noticeable.

Table 3. Precision, recall and F1-score of AlexNet model and FT-AlexNet model

Car manufacturer company	Car model	Precision		Recall		F1-score	
		AlexNet	FT-AlexNet	AlexNet	FT-AlexNet	AlexNet	FT-AlexNet
Chevrolet	Impala	0.58	0.85	0.72	0.80	0.64	0.82
	Silverado	0.49	0.68	0.51	0.82	0.50	0.74
Dodge	Ram	0.74	0.84	0.71	0.76	0.73	0.80
Ford	F150	0.65	0.78	0.57	0.86	0.60	0.82
GMC	Sierra	0.67	0.75	0.64	0.70	0.65	0.73
Honda	Accord	0.66	0.75	0.74	0.89	0.70	0.85
	Civic	0.58	0.82	0.48	0.69	0.53	0.75
Nissan	Altima	0.70	0.80	0.69	0.91	0.70	0.85
Toyota	Camry	0.65	0.95	0.58	0.80	0.62	0.87
	Corolla	0.70	0.85	0.75	0.87	0.72	0.86

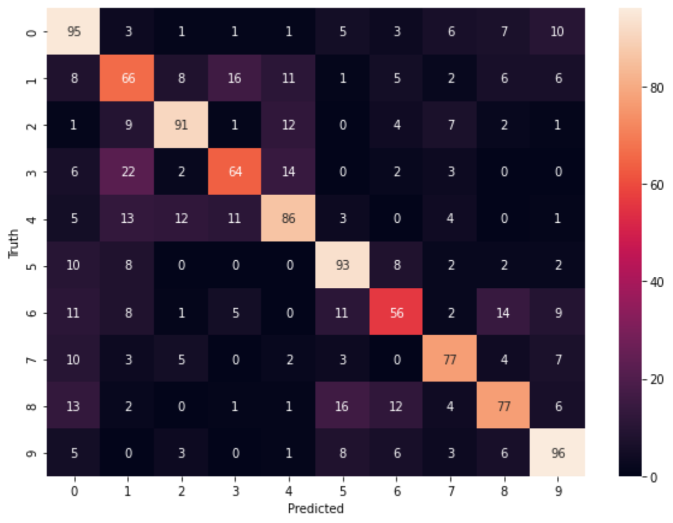

Figure 7. Confusion matrix of AlexNet

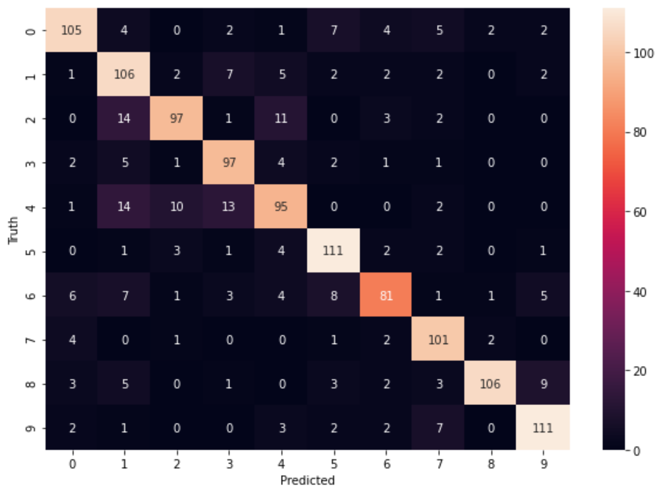

Figure 8. Confusion matrix of FT-AlexNet

4. CONCLUSION

The demand for computer vision-based technology and work on it is increasing significantly. However, in that proportion, CNN-based work with car model recognition has been much less. The computer vision community slightly neglects the field of CMMR. Car model recognition has excellent value in many engineering applications. CMMR is essential for designing intelligent transportation methods such as monitoring or traffic law enforcement procedures. The FT-AlexNet model has experimented with a dataset of 5,000 images. In the future, a larger dataset can be used, which will boost the ability of FT-AlexNet. This paper introduces an improved architecture of AlexNet called FT-AlexNet to recognize cars in our dataset. The FT-AlexNet model has two extra convolutional layers and a max-pooling layer for better accuracy. Several performance matrices were used in this experiment to measure the improvement. Among those matrices, accuracy fits best to understand the model's novelty. By training and testing the model, the result shows that improved FT-AlexNet has outperformed AlexNet.

REFERENCES

[1] I. Fomin, I. Nenahov, and A. Bakhshiev, "Hierarchical system for car make and model recognition on image using neural networks," May 2020, doi: 10.1109/ICIEAM48468.2020.9112026.

[2] F. Tafazzoli, H. Frigui, and K. Nishiyama, "A large and diverse dataset for improved vehicle make and model recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Jul. 2017, vol. 2017-July, pp. 874–881, doi: 10.1109/CVPRW.2017.121.

[3] Q. Bu, S. Lan, and P. Xu, "A CNN based car model recognition improvement," in *ACM International Conference Proceeding Series*, Jun. 2017, vol. Part F1285, pp. 86–89, doi: 10.1145/3094243.3094258.

[4] H. C. Sperker and A. Henrich, "Feature-based object recognition - a case study for car model detection," in *Proceedings - International Workshop on Content-Based Multimedia Indexing*, Jun. 2013, pp. 127–130, doi: 10.1109/CBMI.2013.6576568.

[5] W. Burger and M. J. Burge, "Scale-invariant feature transform (SIFT)," in *Digital Image Processing*, Springer International Publishing, 2022, pp. 709–763.

[6] G. Pearce and N. Pears, "Automatic make and model recognition from frontal images of cars," in *2011 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Aug. 2011, pp. 373–378, doi: 10.1109/AVSS.2011.6027353.

[7] D. T. Munroe and M. G. Madden, "Multi-class and single-class classification approaches to vehicle model recognition from images," in *16th Irish Conference on Artificial Intelligence and Cognitive Science*, 2005, pp. 93–102.

[8] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, Jun. 2008, doi: 10.1016/j.cviu.2007.09.014.

[9] A. J. Siddiqui, A. Mammeri, and A. Boukerche, "Real-time vehicle make and model recognition based on a bag of SURF features," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 11, pp. 3205–3219, Nov. 2016, doi: 10.1109/TITS.2016.2545640.

[10] M. Bansal, M. Kumar, and M. Kumar, "2D object recognition: a comparative analysis of SIFT, SURF and ORB feature descriptors," *Multimedia Tools and Applications*, vol. 80, no. 12, pp. 18839–18857, Feb. 2021, doi: 10.1007/s11042-021-10646-0.

[11] W. Zhou, S. Gao, L. Zhang, and X. Lou, "Histogram of oriented gradients feature extraction from raw bayer pattern images," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 5, pp. 946–950, May 2020, doi: 10.1109/TCSII.2020.2980557.

[12] B. Zhang, "Reliable classification of vehicle types based on cascade classifier ensembles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 322–332, Mar. 2013, doi: 10.1109/TITS.2012.2213814.

[13] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967, doi: 10.1109/TIT.1967.1053964.

[14] D. A. Pisner and D. M. Schnyer, "Support vector machine," in *Machine Learning: Methods and Applications to Brain Disorders*, Elsevier, 2019, pp. 101–121.

[15] J. M. Bernando and A. F. Smith, *Bayesian theory*, vol. 405. Singapore: John Wiley & Sons, Ltd, 2009.

[16] M. M. Arzani and M. Jamzad, "Car type recognition in highways based on wavelet and contourlet feature extraction," in *Proceedings of the 2010 International Conference on Signal and Image Processing, ICSIP 2010*, Dec. 2010, pp. 353–356, doi: 10.1109/ICSIP.2010.5697497.

[17] L. Yang, P. Luo, C. C. Loy, and X. Tang, "A large-scale car dataset for fine-grained categorization and verification," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2015, vol. 07-12-June-2015, pp. 3973–3981, doi: 10.1109/CVPR.2015.7299023.

[18] H. Huttunen, F. S. Yancheshmeh, and C. Ke, "Car type recognition with deep neural networks," in *IEEE Intelligent Vehicles Symposium, Proceedings*, Jun. 2016, vol. 2016-Augus, pp. 1115–1120, doi: 10.1109/IVS.2016.7535529.

[19] Y. Gao and H. J. Lee, "Vehicle make recognition based on convolutional neural network," in *2015 IEEE 2nd International Conference on InformationScience and Security (ICISS)*, Dec. 2015, pp. 1–4, doi: 10.1109/ICISSEC.2015.7371039.

[20] X. Wang, W. Zhang, X. Wu, L. Xiao, Y. Qian, and Z. Fang, "Real-time vehicle type classification with deep convolutional neural networks," *Journal of Real-Time Image Processing*, vol. 16, no. 1, pp. 5–14, Aug. 2019, doi: 10.1007/s11554-017-0712-5.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 2, pp. 1097–1105, 2012.

[22] K. Simonyan and Z. Andrew, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015, p. 14.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2016, vol. 2016-December, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[24] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2015, vol. 07-12-June-2015, pp. 1–9, doi: 10.1109/CVPR.2015.7298594.

[25] "The car connection," *thecarconnection.com*. https://www.thecarconnection.com (accessed Sep. 20, 2021).

[26] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5455–5516, Apr. 2020, doi: 10.1007/s10462-020-09825-6.

[27] K. Fukushima, "A neural network model for selective attention in visual pattern recognition," *Biological Cybernetics*, vol. 55, no. 1, pp. 5–15, Oct. 1986, doi: 10.1007/BF00363973.
[28] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into Imaging*, vol. 9, no. 4, pp. 611–629, Jun. 2018, doi: 10.1007/s13244-018-0639-9.
[29] Y. Gao, W. Liu, and F. Lombardi, "Design and implementation of an approximate softmax layer for deep neural networks," in *Proceedings - IEEE International Symposium on Circuits and Systems*, Oct. 2020, vol. 2020-October, doi: 10.1109/iscas45731.2020.9180870.
[30] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," *arXiv preprint*, p. 12, 2015.

BIOGRAPHIES OF AUTHORS

Md. Asifur Rahman Khan is working as a student at the Department of Software Engineering at Daffodil International University in Bangladesh. He pursued his B.Sc. in Software Engineering in 2021 from Daffodil International University. Currently, he is working as a data analyst in a software company. He also has experience in software development. His research interests are machine learning, deep learning, and computer vision. He can be contacted at email: asifur35-2259@diu.edu.bd.

Raju Talukder is working as a student in the Department of Software Engineering, Daffodil International University, Bangladesh. He is pursuing his B.Sc. in Software Engineering, Major in Cyber Security at Daffodil International University. He worked as a research assistant at the Cyber Security Center, Department of Software Engineering, Daffodil International University, Bangladesh. He is a Certified Ethical Hacker by EC-Council. He is interested in cyber security, machine learning, and vulnerability assessment. He can be contacted at email: raju35-2222@diu.edu.bd.

Md. Anwar Hossen is working as an Assistant Professor at the Department of Software Engineering at Daffodil International University in Bangladesh. He is currently pursuing his Ph.D. from the Department of Computer Science, Iowa State University, USA. He completed his B.Sc. and M.Sc. from the Department of Computer Science and Engineering, Jagannath University, Bangladesh. He was awarded the National Science and Technology (NST) fellowship (2016-2017) by the Ministry of Science and Technology, Bangladesh. His research interests are machine learning, deep learning, computer vision, and intelligent agent design. He can be contacted at email: anwar.swe@diu.edu.bd.

Nusrat Jahan is an Assistant Professor and Head, Department of Information Technology and Management, Daffodil International University, Bangladesh. She is currently pursuing her Ph.D. from the School of Computer and Communication Engineering, Universiti Malaysia Perlis (UniMAP). She completed her M.Sc. and B.Sc. in IT from the Institute of Information Technology, Jahangirnagar University and B.Sc. in Software Engineering from Daffodil International University. She is interested in technology management, computer networks and machine learning. She can be contacted at email: nusrat.swe@diu.edu.bd.