Journal of Computer Engineering, Electronics and Information Technology (COELITE)

Journal homepage: https://ejournal.upi.edu/index.php/COELITE

Performance Evaluation of the Transfer Learning Model in Car Vehicle Detection

*Dafa Fidini Asqav[1, *], R. Rahmawati[2], Monica Marito Manurung[3]*

[1]Department of Computer Engineering, Institut Teknologi Sepuluh Nopember Surabaya, Indonesia
[2]Departement of AI Research, PT Bisa Artificial Indonesia, Indonesia
[3]Departement of Information Engineering, Universitas Mikroskil, Indonesia
Correspondence: E-mail: fidini.18072@mhs.its.ac.id

ABSTRACT

Congestion is a common problem that occurs in big cities. It is necessary to make an ITS (Intelligent Transportation System) using computer vision to monitor street density by counting the number of passing vehicles. The basis of vehicle counting is vehicle detection. This study will compare the performance of vehicle detection models in order to make it easier to determine which model is suitable for implementation. The dataset used is the Vehicle Detection Image Set. The models used are InceptionV3, Xception, MobileNet, MobileNet V2, VGG 16, VGG 19, Efficientnet (B7), and Efficientnet V2 (L). The results show that Inception, Xception, Mobilenet, and VGG19 are the models with the highest test accuracy. The VGG 16 model has the shortest training duration while the Efficientnet V2 (L) has the longest training duration.

ARTICLE INFO

Article History:
Submitted/Received 06 Feb 2023
First Revised 14 Feb 2023
Accepted 11 Mar 2023
First Available online 01 Apr 2023
Publication Date 01 Apr 2023

Keyword:
EfficientNet,
MobileNet,
Transfer Learning,
Vehicle Detection,
VGG.

1 INTRODUCTION

Congestion is a common problem that occurs in big cities today. An imbalance between the quantity and size of existing roads and the increasing number of vehicles from year to year is one of the causes of this problem. The Central Statistics Agency (BPS) notes that in 2022 there will be 143,797,227 units of motorized vehicles where this number will continue to grow (see this link below):
https://www.bps.go.id/indikator/indikator/view_data_pub/0000/api_pub/V2w4dFkwdFNLNU5mSE95Und2UDRMQT09/da_10/1.

Meanwhile, in the same year, BPS noted that the total length of roads in Indonesia was 548,423 km (see https://www.bps.go.id/). Based on data compiled by Tomtom (see https://www.tomtom.com/traffic-index/jakarta-traffic/), Jakarta is the 46th most congested city in the world in 2021 with a congestion rate of 34%. If you look at the previous 2 years, this figure is the smallest number. In 2019 the level of congestion in Jakarta reached 53%, while in 2020 it decreased to 36%.

Various policies have been put in place to overcome this problem, one of which is a traffic monitoring policy to monitor the level of traffic density by counting each vehicle on the road. The conventional method of counting vehicles in traffic monitoring is done using sensors, such as pneumatic tubes, magnetic sensors, inductive loops, etc [1]. However, this conventional method requires a fairly high cost coupled with the cost of maintenance. In addition, this method is difficult to install and less flexible. Therefore, it is necessary to create an Intelligent Transportation System (ITS) using computer vision to calculate vehicles on the highway [2]. One of the basics in ITS is the vehicle detection process. Before vehicles are counted and their density determined, the system first detects whether these objects include vehicles such as cars, trucks, buses, etc. or non-vehicles.

Research conducted by H. Song et al. [3] regarding Vision based Vehicle Detection and Counting System using Deep Learning in Highway Scenes. In this study using a dataset created by researchers taken from video monitoring of highways in Hangzhou, China. The dataset consists of 3 categories, namely cars, buses and trucks with a total of 11,129 images. First of all, the image will be extracted and segmented using Gaussian Mixture Modeling. The vehicle detection process uses YOLOv3 Darknet-53. The results showed that the method created could produce a precision value of 88%, 89% recall, average IOU 71.32%, and mAP 87.88%. This method has a price that is not too high and high stability.

Research conducted by Y. Chen and Z. Li [4] regarding An Effective Approach of Vehicle Detection Using Deep Learning. In this study using the BDD100K dataset consisting of six categories, namely cars, buses, pedestrians, trucks, motorbikes, and bicycles. The model used is YOLOv3. Results show that the model has a MAP value of 72.8. The model can detect cars well but for the bus category it is slightly poor and it is performing very poorly in the truck category.

Research conducted by [5] regarding Vehicle Type Detection Based on Deep Learning in Traffic Scene. In this study using the MIT and Clatech vehicle dataset with the ZF and VGG16 models. Samples will be trained using the RPN network which will then be entered into the Fast-RCNN network until the network converges. The results show that the ZF model with a total of 3052 data obtained the accuracy values for each category for buses, minibuses and SUVs of 79.9%, 73.9% and 68.3%. Whereas the VGG16 model with the same amount of data and categories obtained accuracy values of 82.3%, 74.8% and 70.1%. In the ZF model with a total of 5042 data, the accuracy values for each category for buses, minibuses and SUVs were

84.0%, 81.0% and 73.7%. Whereas the VGG16 model with the same amount of data and categories obtained accuracy values of 84.4%, 83.8% and 78.3%.

Research conducted by [6] regarding A Multi-Task Faster RCnn Method for 3d Vehicle Detection Based on A Single Image. This study uses the KITTI dataset using the Faster R-CNN and RPN methods. Faster R-CNN is used as the base network and ResNet-50 backbone network. The results show that the method is feasible and can be used in vehicle detection, especially in autonomous driving vehicles.

Research conducted by A. Wang [7] regarding the Vehicle Recognition Algorithm Based on Improved YOLOv3. The dataset used is video monitoring. In this study using the YOLOv3 method with the Darknet-53 model. YOLOv3 will be enhanced by using Dpow3x332 and residual blocks to perform feature extraction. The results show that after the method is improved, the loss value can be seen in distinguishing large and small vehicles. The loss value is 0.0006 for small vehicles and 0.00012 for large vehicles which is ¼ times smaller than the previous method.

The architecture used in this study is the Convolutional Neural Network (CNN) using InceptionV3, Xception, MobileNet, MobileNet V2, VGG 16, VGG 19, Efficientnet (B7), and Efficientnet V2 (L) as the model. The dataset used is the Vehicle Detection Image Set. This dataset consists of 17760 images containing both vehicles and non-vehicles. This study contributes to comparing the level of accuracy of each model. Readers will be able to analyze various existing models to determine which model is best for the research they are conducting. This research is useful as a basis for further research as well as a basis for other research related to. In addition, the results of this study can be implemented and integrated with existing vehicle detection hardware in everyday life. This paper starts with an Introduction which explains the background of the problem, objectives, objectivity, and literature review Then proceed with the method which discusses the stages of research. Third, there are Results and Discussion which discuss the results of the research and discuss them with several previous studies. Finally, the last section contains the conclusions of the research that has been carried out from beginning to end.

2 METHODS

In general, the method of this research consists of several steps including Data Acquisition, Splitting Dataset, Data Augmentation, Model Training, Model Testing, and Evaluation as shown in **Figure 1**. Training uses a computer with GPU RTX 2080 SUPER, Intel i7 9700K CPU, and 32GB RAM.

2.1 Data Acquisition

At this stage, the data collection process is carried out to create a dataset. However, this time the researchers did not manually create datasets but used public datasets that are already available on the internet. The dataset used is the Vehicle Detection Image Set obtained from Kaggle (see https://www.kaggle.com/datasets/brsdincer/vehicle-detection-image-set). This dataset consists of 17760 images containing both vehicles and non vehicles. The dataset consists of 2 classes, namely the non vehicles class and the vehicles class. In the non-vehicles class there are 8968 pictures and in the vehicles class there are 8792 pictures. The outline of the whole project described in **Figure 1.**, which include their parts: knowledge-database provided by experts in different fields; software-agent uploaded by developers who access to our server; and client-part which usually occur in factories.

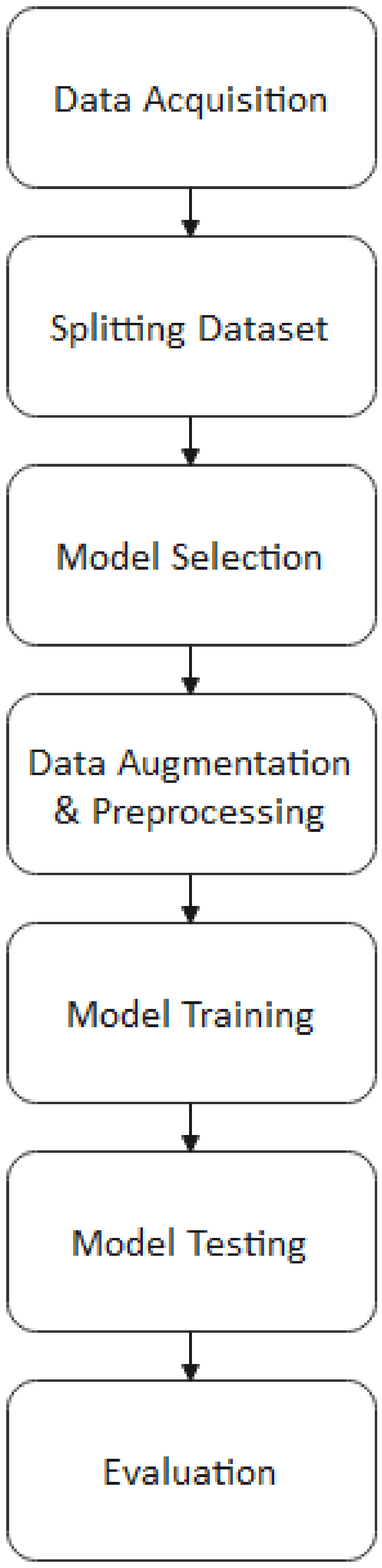

Figure 1. Research Flowchart.

2.2 Splitting Dataset

The dataset will be divided into 2 parts, namely the training set and the testing set using the train test split function in the Scikit-learn library. As much as 90% of the data will be used for training data and 10% for testing data.

2.3 Model Selection

At this stage the researcher chooses which models will be used. There are 8 models that will be used. The model is obtained from the transfer learning model available in tensorflow. These models include.

2.3.1 Inception V3

Specifically, the Inception V3 model includes three parts: basic convolutional block, enhanced Inception module, and classifier. The basic convolutional block, which replaces the convolutional with a max-pooling layer, is used for feature extraction. The upgraded Inception module is designed on the basis of Network In-Network, where multiscale convolutions are performed in parallel and the convolution results of each branch are further combined. The use of a classifier makes the training results more stable and the gradient convergence obtained is better and the problem of overfitting is reduced [8][9].

2.3.2 Xception

Xception is part of the CNN architecture which is based on deep separable convolution layers. Xception is an acronym for Extreme Inception. The Xception architecture consists of 36 convolutional layers. The 36 convolutional layers are organized into 14 modules, all of which have linear residual connections around them, except for the first and last modules [10][11].

2.3.3 MobileNet

MobileNet is based on convolution layers which can be separated in depth except for the first layer which is full convolution layer [12-14]. In total there are 28 layers in this architecture. All layers are followed by a batchnorm and non-linear ReLU except for the last layer which is fully connected and included in the softmax layer for classification.

2.3.4 MobileNet V2

Process on MobileNet V2 starts with feature extraction using Conv2D with 3x3 kernel. The results from Conv2D will enter the bottleneck layer. The bottleneck layer consists of 3 convolution processes, namely 1x1 convolution, 3x3 depthwise convolution, and 1x1 pointwise convolution [12-15]. In this study, the architecture used is the same as the MobileNet V2 architecture in general, but Flatten and Dropout layers are added to reduce the level of overfitting. In addition, for the last layer, the activation function that originally used softmax was changed to sigmoid because the dataset is a binary class, not a multiclass.

2.3.5 VGG 16

VGG 16 architecture consists of a convolutional layer, max pooling layer, fully connected layer, and softmax layer [16][17]. VGG 16 uses ReLU for its activation function. In this study the architecture used is the same as the VGG 16 architecture in general but added Flatten and Dropout layers to reduce the level of overfitting. In addition, for the last layer, the activation function that originally used softmax was changed to sigmoid because the dataset is a binary class, not a multiclass.

2.3.6 VGG 19

VGG 19 architecture consists of 19 layers with 16 convolutional layers, 3 fully connected layers, 5 max pooling layers and 1 softmax layer [16-18]. In this study the architecture used is the same as the VGG 19 architecture in general but added Flatten and Dropout layers to reduce the level of overfitting. In addition, for the last layer, the activation function that originally used softmax was changed to sigmoid because the dataset is a binary class, not a multiclass

2.3.7 EfficientNet (B7)

The EfficientNet architecture is based on the reverse bottleneck MBConv architecture previously introduced in MobileNet V2. It's called an inverted bottleneck because the incoming MBConv block consists of layers that expand then shrink like a compressed channel. This design contains deep convolutions [19][20]. In this study, the architecture used is the same as the EfficientNet architecture in general but added Flatten and Dropout layers to reduce the level of overfitting. In addition, for the last layer, the activation function that originally used softmax was changed to sigmoid because the dataset is a binary class, not a multiclass.

2.3.8 EfficientNet V2

The EfficientNet V2 architecture uses MBConv and adds fused-MBConv to the initial layer. EfficientNet V2 has a smaller expansion ratio and kernel size but has many layers [21]. In this study the architecture used is the same as the EfficientNet V2 architecture in general but added Flatten and Dropout layers to reduce the level of overfitting. In addition, for the last layer, the activation function that originally used softmax was changed to sigmoid.

2.4 Data Augmentation and Preprocessing

The data augmentation stage aims to increase the amount of training data so that the existing data is more diverse. Each image will be flipped horizontally and vertically. In addition, the image will be rotated by 20° and rescaled with a ratio of 1./255. At the data preprocessing stage, each label in the image will be converted into a numeric using a label encoder.

2.5 Model Training, Testing, Evaluation

Each existing model will be trained using training data. The model will be trained sequentially until the model produces a good accuracy value. After the model is trained, then model testing will be carried out using testing data. Models that have been trained and tested will be evaluated. Evaluation of the model uses several parameters, namely the value of accuracy, precision, recall, f1-score, and AUC.

3 RESULTS AND DISCUSSION

From the experimental results in this study, the results obtained from the model were evaluated. The results obtained are accuracy, recall, precision, F1, and AUC, and training duration for each model, as shown in **Figure 2**.

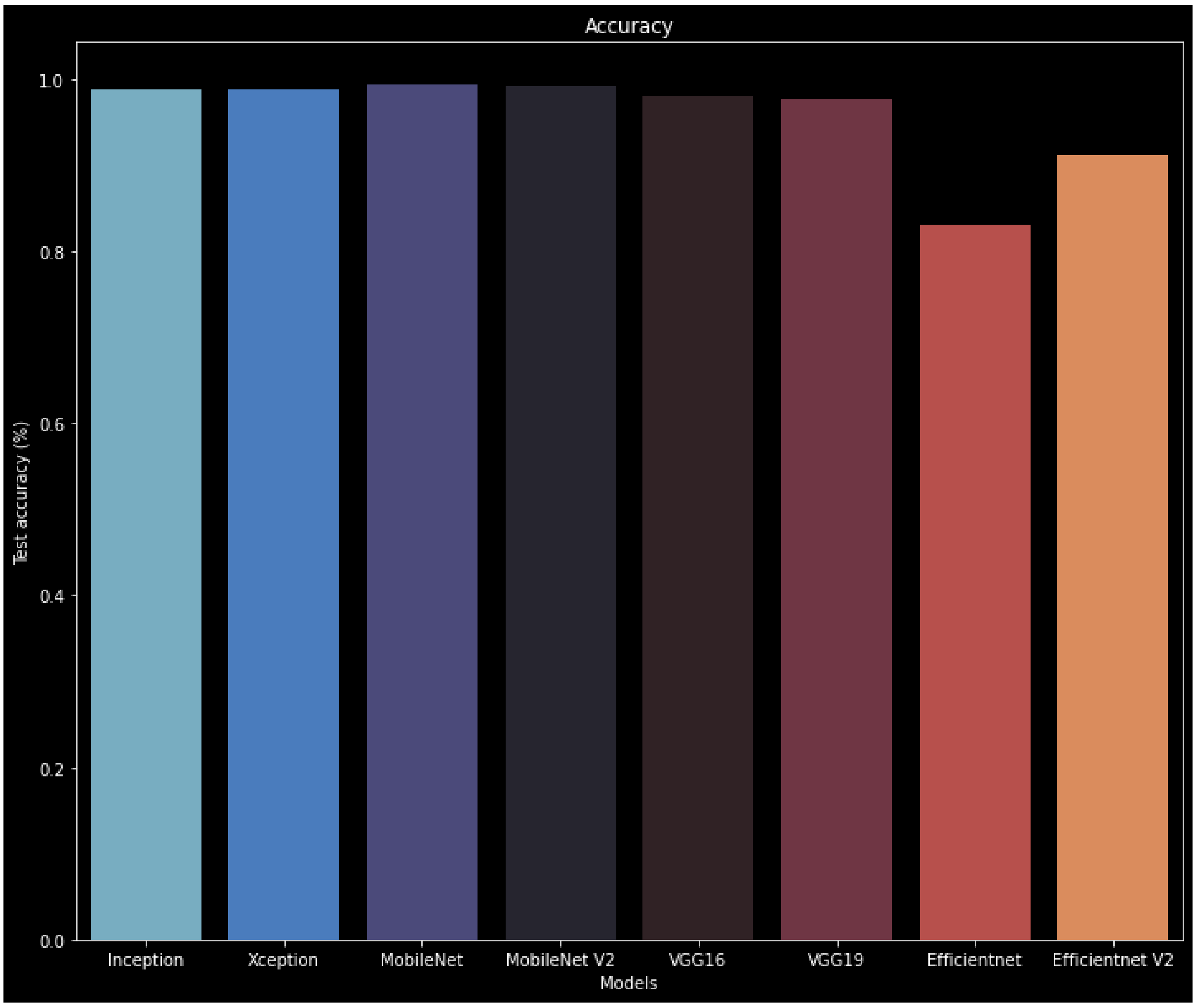

Figure 2. Graph of Comparison of Accuracy Values Between Models.

Inception, Xception, Mobilenet, and VGG19 are the models with the highest test accuracy. Mobilenet V2, Efficientnet (B7), and Efficientnet V2 (L) experience overfitting with training accuracy reaching 98% and above but test accuracy is far below that value. The VGG16 and VGG19 models have poor accuracy during training but the results on test accuracy are much better. Mobilenet V2 and VGG16 are the fastest models in completing training.

VGG16 was the fastest model in training with a training time of 02:12, then followed by VGG19 with 04:04 minutes, MobileNet V2, with 03:39 minutes, MobileNet with 07:17 minutes, Inception with 07:23 minutes, and Xception in 07:23 min. Efficientnet (B7) and Efficientnet V2 (L) are the longest models with a training time of 51:14 minutes and 01:28:41 hours.

Efficientnet V2 (L) has a Recall value of 29%. Efficientnet (B7) has better performance but not perfect with a Recall value of 76%. VGG 16 also has a Recall value that is less than perfect at 78%. MobileNet V2 has a good Recall value with a value of 99% but a bad precision value with a value of 55%, as shown in **Figure 3**.

Table 1. Table of accuracy values and length of training.

Model	Training Accuracy (%)	Test Accuracy (%)	Recall (%)	Precision (%)	F1 (%)	Training Time (Hours)
Inception	0.976870	0.985360	0.989819	0.980942	0.985360	0:07:23
Xception	0.987418	0.981419	0.988688	0.974359	0.981471	0:08:43
MobileNet	0.978213	0.972973	0.990950	0.956332	0.973333	0:07:17
MobileNet V2	0.997266	0.992117	0.998869	0.555346	0.713824	0:03:39
VGG16	0.573127	0.894144	0.789593	0.997143	0.881313	0:02:12
VGG19	0.720506	0.972973	0.990950	0.956332	0.973333	0:04:04
Efficientnet (B7)	0.991751	0.878378	0.764706	0.988304	0.862245	0:51:14
Efficientnet V2 (L)	0.988911	0.650901	0.298643	1.000000	0.459930	1:28:41

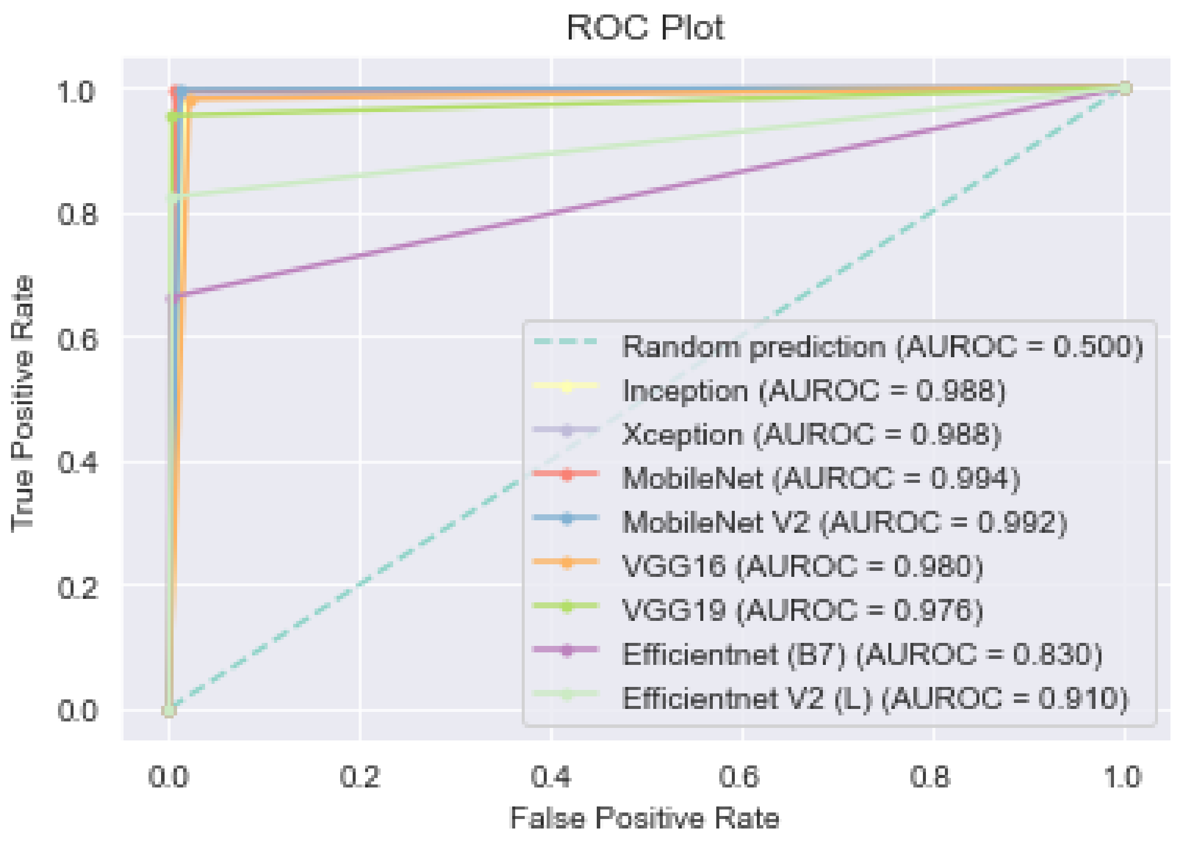

Figure 3. ROC Every Model.

From the picture above it can be seen that the two models Efficientnet has a False Negative case on 3 True vehicles images. VGG16 and Mobilenet V2 also have a False Positive case on one of the different True non vehicles images. The Inception, Xception, Mobilenet, and VGG 19 models have no False Positive or False Negative cases in the image above.

Based on the test results above, each model has different training duration and accuracy values. There is no correlation between training duration and accuracy. This is because it can be seen that some models with a short training duration have a high accuracy value, while others have a fairly low accuracy value. Likewise with models with a long enough duration, some have high accuracy values, some are quite low. As shown in **Figure 4**.

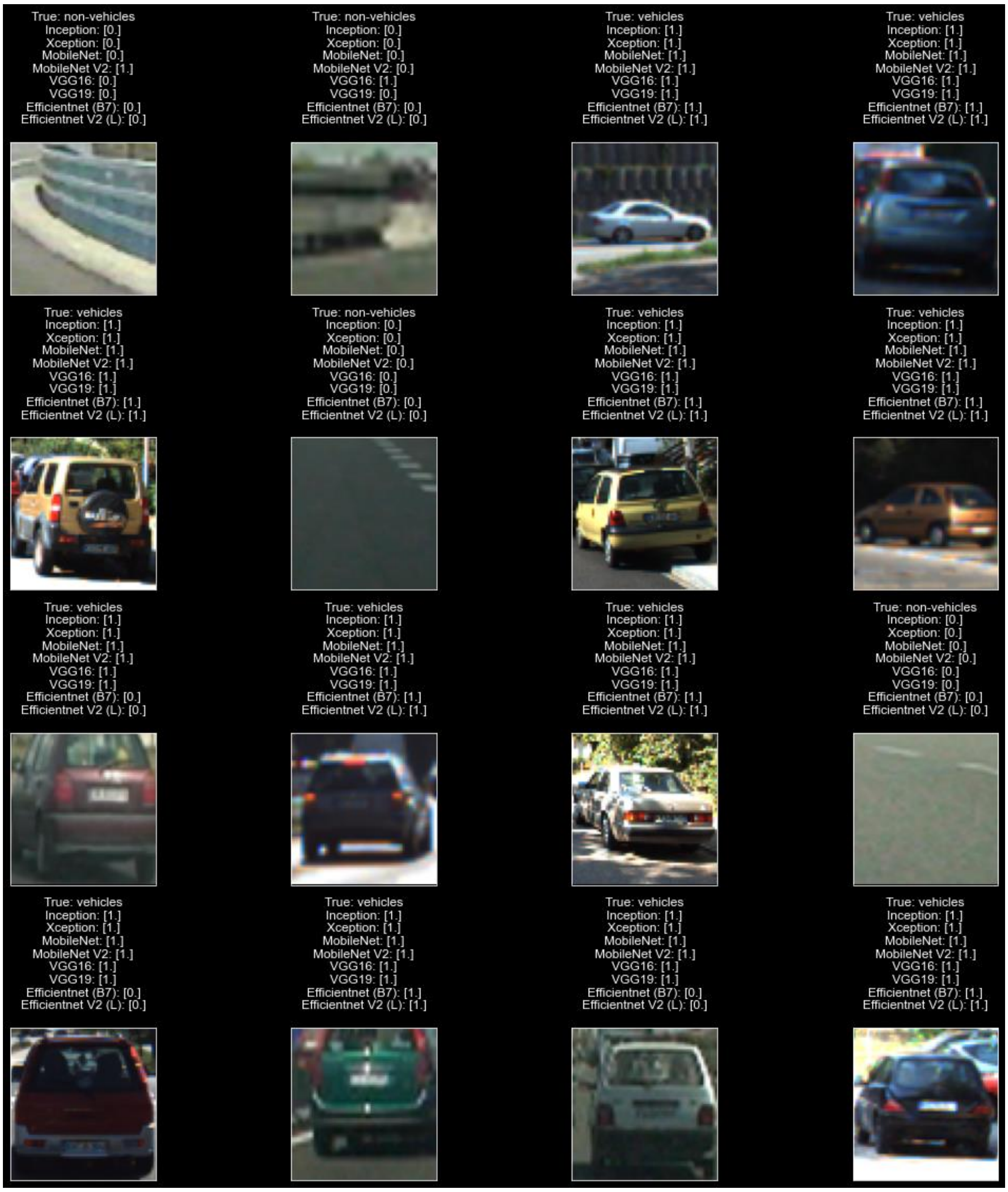

Figure 4. Testing Model.

Models other than Efficientnet (B7) and Efficientnet V2 (L) have a short training time because the model accepts a lower resolution than the Efficientnet model. Models that have a fairly good accuracy value and also have a shorter training time have many advantages. The relatively short training time makes inferences faster so that they are easier to apply in small devices that can be used in the field, such as CCTV cameras. Another advantage is that the model iteration is faster and easier. Faster model training will make it easier to iterate changes and optimize model performance.

The InceptionV3, Xception, MobileNet, MobileNet V2, VGG 16, and VGG 19 models were introduced earlier than the Efficientnet (B7) and Efficientnet V2 (L) models. Therefore, it can be seen that the 5 models have a fairly good level of accuracy as seen in the InceptionV3, VGG 16, and VGG 19 models, which experienced an increase in the accuracy value during testing when compared to the accuracy value during training.

Even though Xception, MobileNet, and MobileNet V2 experienced a decrease in the level of accuracy during testing. This is different from the Efficientnet (B7) and Efficientnet V2(L) models. This model was introduced recently, namely in 2019 and 2021 by M Tan with the advantage of a faster training duration.

However, this is inversely proportional to the results of our tests. Although the Efficientnet (B7) and Efficientnet V2 (L) models have quite good training accuracy values, they have quite a long training duration and worse test accuracy values than the other models.

Even though the Efficientnet (B7) and Efficientnet V2 (L) models have quite good training accuracy values, they have quite a long training duration compared to the other models. To find out the causes of the performance of the Efficientnet (B7) and Efficientnet V2 (L) models, which have a long testing duration but poor accuracy results, this research cannot yet be proven because they are outside the scope of the research.

4 CONCLUSION

In our project, we use the vibration monitoring method to make the machine diagnosis. Some energy harvesters are used to collect power from rotation machines, and our laboratory constructed an autonomous vibration sensor that consumes the power collected by these energy harvesters instead of external power supply, such as batteries and AC power supply. To implement the planning agent, we use PDDL (Planning Domain Description Language) technique and utilize the inference engine of PPDL to do the planning jobs.

To implement the planning agent, we use the PDDL technique to set up the inheritance relationship among different agents. After involving the inheritance feature, it can improve the efficiency of searching and extendibility of our multi-agent system.

5 AUTHORS' NOTE

The authors declare that there is no conflict of interest regarding the publication of this article. Authors confirmed that the paper was free of plagiarism.

6 REFERENCES

[1] Li, S., Chang, F., and Liu, C. (2020). Bi-directional dense traffic counting based on spatio-temporal counting feature and counting-LSTM network. *IEEE Transactions on Intelligent Transportation Systems, 22(*12), 7395-7407.

[2] Azimjonov, J., and Özmen, A. (2021). A real-time vehicle detection and a novel vehicle tracking systems for estimating and monitoring traffic flow on highways. *Advanced Engineering Informatics, 50*(1), 1-14.

[3] Song, H., Liang, H., Li, H., Dai, Z., and Yun, X. (2019). Vision-based vehicle detection and counting system using deep learning in highway scenes. *European Transport Research Review, 11*(1), 1-16.

[4] Chen, Y., and Li, Z. (2022). An effective approach of vehicle detection using deep learning. *Computational Intelligence and Neuroscience, 2022*(1), 1-9.

[5] Li, Y., Song, B., Kang, X., Du, X., and Guizani, M. (2018). Vehicle-type detection based on compressed sensing and deep learning in vehicular networks. *Sensors, 18*(12), 1-15.

[6] Arora, N., Kumar, Y., Karkra, R., and Kumar, M. (2022). Automatic vehicle detection system in different environment conditions using fast R-CNN. *Multimedia Tools and Applications, 81*(13), 18715-18735.

[7] Gai, W., Liu, Y., Zhang, J., and Jing, G. (2021). An improved tiny YOLOv3 for real-time object detection. *Systems Science and Control Engineering, 9*(1), 314-321.

[8] Al Husaini, M. A. S., Habaebi, M. H., Gunawan, T. S., Islam, M. R., Elsheikh, E. A., and Suliman, F. M. (2022). Thermal-based early breast cancer detection using inception V3, inception V4 and modified inception MV4. *Neural Computing and Applications, 34*(1), 333-348.

[9] Lin, C., Li, L., Luo, W., Wang, K. C., and Guo, J. (2019). Transfer learning based traffic sign recognition using inception-v3 model. *Periodica Polytechnica Transportation Engineering, 47*(3), 242-250.

[10] Chen, B., Liu, X., Zheng, Y., Zhao, G., and Shi, Y. Q. (2021). A robust GAN-generated face detection method based on dual-color spaces and an improved xception. *IEEE Transactions on Circuits and Systems for Video Technology, 32*(6), 3527-3538.

[11] Polat, Ö. Z. L. E. M. (2021). Detection of COVID-19 from chest CT images using xception architecture: a deep transfer learning based approach. *Sakarya University Journal of Science (SAUJS), 25*(3), 813-823.

[12] Jing, J., Wang, Z., Rätsch, M., and Zhang, H. (2022). Mobile-unet: an efficient convolutional neural network for fabric defect detection. *Textile Research Journal, 92*(1), 30-42.

[13] Kulkarni, U., Meena, S. M., Gurlahosur, S. V., and Bhogar, G. (2021). Quantization friendly mobilenet (QF-mobilenet) architecture for vision based applications on embedded platforms. *Neural Networks, 136*(1), 28-39.

[14] Ghosh, T., Abedin, M. M. H. Z., Chowdhury, S. M., Tasnim, Z., Karim, T., Reza, S., Saika, S., and Yousuf, M. A. (2020). Bangla handwritten character recognition using MobileNet V1 architecture*. Bulletin of Electrical Engineering and Informatics, 9*(6), 2547-2554.

[15] Wibowo, A., Adhi Hartanto, C., and Wisnu Wirawan, P. (2020). Android skin cancer detection and classification based on MobileNet V2 model. *International Journal of Advances in Intelligent Informatics, 6*(2), 135-148.

[16] Pardede, J., Sitohang, B., Akbar, S., and Khodra, M. L. (2021). Implementation of transfer learning using VGG16 on fruit ripeness detection. *International Journal of Intelligent Systems and Applications, 13*(2), 52-61.

DOI: https://doi.org/10.17509/coelite.v2i1.59716
p- ISSN 2829-4157 e- ISSN 2829-4149

[17] Rangarajan, A. K., and Purushothaman, R. (2020). Disease classification in eggplant using pre-trained VGG16 and MSVM. *Scientific Reports, 10*(1), 1-11.
[18] Mateen, M., Wen, J., Nasrullah, N., Song, S., and Huang, Z. (2018). Fundus image classification using VGG-19 architecture with PCA and SVD. *Symmetry, 11*(1), 1-12.
[19] Chowdhury, M. E., Rahman, T., Khandakar, A., Ayari, M. A., Khan, A. U., Khan, M. S., Al-Emadi, N., Ibne Reaz, M. B., Islam, M. T., and Ali, S. H. M. (2021). Automatic and reliable leaf disease detection using deep learning techniques. *AgriEngineering, 3*(2), 294-312.
[20] Ali, K., Shaikh, Z. A., Khan, A. A., and Laghari, A. A. (2022). Multiclass skin cancer classification using EfficientNets–a first step towards preventing skin cancer. *Neuroscience Informatics, 2*(4), 1-10.
[21] Rizal, S., Ibrahim, N., Pratiwi, N. K. C., Saidah, S., and Fu'adah, R. Y. N. (2020). Deep learning untuk klasifikasi diabetic retinopathy menggunakan model efficientnet. *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, and Teknik Elektronika, 8*(3), 693-705.