Hyperparameter Optimization for Improving Recognition Efficiency of an Adaptive Learning System

DIEM-PHUC TRAN\textsuperscript{1}, GIA-NHU NGUYEN\textsuperscript{1}, (Member, IEEE), AND VAN-DUNG HOANG\textsuperscript{2}

\textsuperscript{1}Graduate School, Duy Tan University, Da Nang 550000, Vietnam
\textsuperscript{2}Faculty of Information Technology, Ho Chi Minh City University of Technology and Education, Ho Chi Minh City 700000, Vietnam

Corresponding author: Van-Dung Hoang (dunghv@hcmute.edu.vn)

ABSTRACT

Today, several studies have been concretized in the areas of robotics, self-driving cars, intelligent assistance systems, and so on. Developing an increasingly optimal neural network in terms of accuracy and processing speed for resource-limited systems has become a major research trend. Some research orientations include focusing on developing solutions to optimize machine learning models and learning parameters. In this study, we investigated an optimization solution for learning hyperparameters of adaptive learning systems for improving object recognition accuracy. The proposed method was developed from a framework searching a set of learning hyperparameters based on the evaluation of the previous CNN model with the collected dataset during the movement of advanced driver assistance systems (ADAS) equipment. The proposed solution consists of some major steps in a loop of adaptive learning system, such as (1) training an initial recognition model, (2) locating and receiving image data of different cases of the object during ADAS movement based on object tracking process, (3) finding optimal hyperparameters on the found dataset based on the previous recognition model, and (4) using the trained recognition model to update the current recognition model. The experimental results proved that the trained recognition model was capable of being more intelligent and displayed more diverse recognition than the previous model. The updated task for the recognition model was continuously repeated throughout the ADAS life. This approach supports and enables the recognition system to be self-adaptive and more intelligent in real life settings without manually processing.

INDEX TERMS

Deep learning, adaptive learning, advanced driver assistance systems, object recognition, optimal learning hyperparameters.

I. INTRODUCTION

Technology is constantly developing with new ideas about artificial intelligence and has created new products with practical value in daily life. Numerous studies have been concretized in the fields of robotics, automation, self-driving cars, intelligent assistance systems, and so on. Developing artificial neural networks capable of intelligent and asymptotic values similar to the human brain has become a major research trend. Thus, convolutional neural network (CNN) models have become a relevant topic due to their capacity for adaptive learning, self-intelligence, coping, and automatically adjusting to diverse specific circumstances.

In the most recent article [1], the adaptive-learning model in object recognition was proposed (Figure 1). Generally, the architecture simulates an on-road ADAS system. Used CNN model is intended to enable the system to detect and recognize surrounding objects such as vehicle, traffic sign, pedestrian, and so on. In case its performance of object recognition with low accuracy due to distance, weather or being obscured, the system immediately makes object tracking, simultaneously storing the image for a dataset establishment. Once the object is recognized with high accuracy, the saved data of system are utilized to train and add "intelligence" to the CNN detection model.

In the adaptive-learning model, there are two basic issues that need to be addressed in adaptive learning, including (1) data optimisation and (2) algorithmic and...
II. RELATED WORKS

Recent studies of ADAS systems have proposed solutions to detection with high accuracy, high speed of training and quick processing. The key solutions include the solution of using sensors and the image processing one. For the solution of using sensors, object detection and tracking like vehicles, pedestrian and other obstacles are focused on [2], [3]. Whereas, the image processing method typically utilizes CNN models, including AlexNet [4], GoogleNet [5], Microsoft ResNet [6], R-CNN [7], Fast R-CNN [8], Faster R-CNN [9], and so on. Based on these models, several new developments to improve the accuracy and enhance the processing speed of the system are now available [10], [11], [12].

Select detection models: These solutions focus on the automatic selection of recognition model types without using a specific default model (e.g., choosing between CNN and SVM) [13], [14]. The selection of training models will make it possible to solve each specific case of the data and provide greater accuracy. In addition, these solutions also allow assessment of data types, data models, and so on for automatic selection of an appropriate model [15], [16]. However, the studies and assessments of these solutions still have several limitations in terms of quantity and quality, and these issues remain unresolved. In addition, a higher hardware configuration is required for a solution during data processing.

Solving problems with data includes developing a model that is capable of self-improving data, enabling a specific CNN model to automatically update data without any human intervention. Adaptive data is a potential research trend where the CNN model can learn and update models so that they resemble the behavior model of the human mind. For adaptive data, there are many studies regarding online tracking that adjust data to track objects or extract features of these objects [15], [17]. Our recent paper [1] proposed using an object tracking process, collecting appropriate data, and automatically retraining the recognition model as a solution to enhance automatic recognition quality of objects. As a result, the post-trained model is more identifiable than the old one. In particular, the model works automatically without any human intervention and adjustment.

Solving problem with algorithms and parameters is solution focusing on algorithms and parameters of CNN models, which creates adaptive changes in CNN layers during training. Specifically, some proposals centralize building frames that are embedded in different layer positions to change weights in the training and recognition process of the CNN model [18]. Other research focus on changing features, including alignment of local depth features to achieve model simplification [13] or changing features between convolutional layers and customizing layers [19], [20], [14]. In general, the research aims to optimize the structure of the CNN model. The changes in layer layout and layer customization have resulted in positive changes. The new models offer a smarter training process, especially in terms of their ability to retain important features during the training [21], [17], [22]–[27]. Recently, several proposals have focused on the automatic selection of training process parameters [17], [21], –[27]. Among them, one solution initially used a small dataset to evaluate the parameters’ effectiveness before selecting the appropriate parameters and conducting training evaluation on the entire data [17]. A different solution used the random search method of parameters before conducting cross-validation on a given number of times to choose parameters matching the recognition model [21]. Furthermore, a combination of evolutionary algorithms to automatically optimize CNN structure by hyperparameters was used in one study [26]. Regarding the configuration
hyperparameter selection of a CNN model, methods using the random search [28], the grid search, or Bayer algorithms [29]–[31] are the most notable.

### III. THE BACKGROUND OF ADAPTIVE LEARNING

In our most recent proposal [1], a model that was adaptive and that automatically updated data was demonstrated (Figure 1).

#### A. BRIEFLY LOOKING BACK AT PREVIOUS APPROACHES

The general idea of the recognition models is based on demonstrating adaptive learning using CNN technology. The models can be applied to different types of objects. However, for convenience in analyzing the proposed method and functional blocks, the problem of traffic signs and vehicle classification was used as an object (hereinafter referred to as Object) to illustrate the idea.

There are two CNN models used in this method, the IONet model for objects recognition (IO recognition) and the PDNet model for confidence determination and objects classification.

Problem description: It is assumed that the two initial CNN models, which are IONet and PDNet with the initial dataset, were trained. During the on-the-road journey, ADAS uses the models to recognize Objects and make appropriate decisions. However, there are some cases in which the system recognizes signs with a low confidence score during processing and recognition. This situation occurs when the system encounters data that is incompatible with the training dataset or when the information is incomplete. The incompatible and jammed data, which is known as hard samples or strange samples (or new samples), are often caused by long distance, signs obscured by other objects, warped or blurred signs, vehicles moving in conditions of low light, rain, snow, motion disturbance, and so on. The adaptive learning model is triggered to recognize new samples. The system will store images with low confidence scores (interest objects - IO) and continue to track (confident tracking) objects. The tracking process aims to identify the following cases:

1. **Lost objects**: Objects initially recognized as low confidence ones, which are tracked through n frames, are still recognized as interest objects with low confidence value and then do not appear in the next frame.

2. **Negative Objects**: Objects initially recognized as interest object but with low confidence scores (less than Confidence$_H$), which are tracked through n frames and which are eventually recognized as not being interest objects. 
   \[ NO = \{O_1, O_2, \ldots O_n | O_i \in IO \text{ and } Conf(O_i) < Confidence_H \text{ where } i = 1, \ldots n - 1 \text{ and } O_n \notin IO \} \]

3. **Positive Objects**: Objects initially recognized as interest objects with low confidence scores (less than Confidence$_H$), which are tracked through n frames, and whose confidence scores eventually reach Confidence$_H$.
   \[ PO = \{O_1, O_2, \ldots O_n | O_i \in IO \text{ and } Conf(O_n) \geq Confidence_H \} \]

Positive Data and Negative Data are bins where Positive Object and Negative Object datasets are stored. When the amount of data in the Positive Datasets and Negative Datasets is large enough, the retraining model CNN task is processed. The newly trained model (PDNet) is selected and evaluated to the previously retrained models, the best of which is used to update the recognition model of the system. The adaptive learning process is ongoing throughout the ADAS process.

#### B. SOME DRAWBACKS THAT NEED TO BE IMPROVED OR SOLVED

The previous proposed ADAS model (Figure 1) has the capacity to self-improve accuracy and become ‘smarter’. However, it was realized at that time that the ADAS system collected and updated new datasets to retrain the model. There was a change in this dataset compared to the previous dataset, whereas the structure of the CNN model during the training and recognition process of the PDNet model remained unchanged. Thus, we propose a solution of automatically changing hyperparameters to train the new CNN model. This method will be adaptive and suitable to each training dataset and is expected to attain higher accuracy. The PDNet model improves the accuracy of object recognition and fully adapt to the change of objects in a specific case.

#### IV. PROPOSED METHOD

The training model and the proposed solution were inherited from the model proposed in [1]. This proposed method makes a new contribution by changing the Retrained PDNet function block as illustrated in Figure 1. The HyperNet [32] function is added to enable the hyperparameter search for the training model, which improves recognition efficiency. The appropriate hyperparameter is automatically solved by the Bayesian approach [33], [23]. The overall proposed method is presented in Figure 2.

Furthermore, the data collected during ADAS movement is constantly changing and refreshed. There is no change in the structural parameters of the CNN model and training parameters in the retraining process of the previous CNN model. Therefore in theory, it is necessary to change the architecture of the CNN model and training parameters to ensure that they match with each new dataset. However, our adaptive solution for retraining the recognition model inherits its ‘intelligence’ from the previous model because of the nature of the proposed solution. Therefore, searching and changing the CNN model’s architecture is not suggested. The solution will focus on finding important hyperparameters of the training process. Then, the most equivalent and optimal model is expected to be found.

#### A. OBJECTS RECOGNITION

For recognition, ADAS uses the IONet recognition model (semantic segmentation model), which was proposed in [1]. IONet is an independent CNN model that remains unchanged during the ADAS performance process. However, discussion of this model is beyond the purview of the current research. When an object is identified by the IONet model...
with a certain reliability, the system will perform tracking and recognition using the PDNet model.

To assess the overall experimental process of the proposed solution, two different groups of objects were experimented on. They included vehicles and traffic signs, as described in Table 1 and Table 2, respectively. The vehicle recognition model is referred to as PDNet-Vehicle, and the recognition model of traffic signs is PDNet-TrafficSign.

**B. PDNET ARCHITECTURE**

The proposed PDNet architecture remained the same as in [1]. Our architecture was constructed based on a series network of CNN, consisting of 27 layers and an input image size of $64 \times 64 \times 3$. Within the algorithm, the proposed method was able to find the PDNet model with the best architecture that was suitable to each dataset found during ADAS movement. However, because of the PDNet model’s inheritance, the architecture of the PDNet remained the same during retraining. In addition, the main advantage of the proposed method was the use of a model with simple architecture that could still attain high accuracy. The adaptive learning process will gradually help the model to increase the accuracy and recognition of a variety of objects. The model’s ‘intelligence’ will increase over time and under ADAS movement. Moreover, the simple model architecture helped the retraining process to be shortened in terms of time. The details of the network are presented in Table 3.

**C. HYPERPARAMETERS SELECTION**

In the retraining process of the CNN model (PDNet model), many hyperparameters were configured, as shown in Table 4. These hyperparameters determined the quality, time, and more of the training process of a CNN model.

In the study, using the Bayesian algorithm was proposed to find adaption and change over six hyperparameters. These included ‘InitialLearnRate’, ‘L2Regularization’, ‘Momentum’, ‘MiniBatchSize’, ‘GradientThreshold’ and ‘GradientThresholdMethod’. These hyperparameters were the important ones that could change and adapting to new datasets.

### Table 1. The object for PDNet model recognition.

| ROI     | Name                  | Description                           |
|---------|-----------------------|---------------------------------------|
| Motorcycle | No Entry             | The traffic sign: No driving beyond this sign |
| Car     | No parking, stopping  | The traffic sign: No parking or stopping beyond this sign |
| Coach   | Non-priority intersection to the right | The traffic sign: Non-priority intersection to the right |
| Truck   |                       |                                       |

### Table 2. The object for PDNet model recognition.

| ROI     | Name                  | Description                           |
|---------|-----------------------|---------------------------------------|
| Motorcycle | No Entry             | The traffic sign: No driving beyond this sign |
| Car     | No parking, stopping  | The traffic sign: No parking or stopping beyond this sign |
| Coach   | Non-priority intersection to the right | The traffic sign: Non-priority intersection to the right |
| Truck   |                       |                                       |

### Table 3. PDNet model structure and parameters.

| Layer | Name             | Description       |
|-------|------------------|-------------------|
| 1     | ‘imageinput’     | 64x64x3 images    |
| 2     | ‘conv_1’         | 128 7x7x3 convolutions |
| 3     | ‘relu_1’         | ReLU              |
| 4     | ‘crossnorm_1’    | Cross channel normalization |
| 5     | ‘maxpool_1’      | 3x3 max pooling   |
| 6     | ‘conv_2’         | 128 7x7x128 convolutions |
| 7     | ‘relu_2’         | ReLU              |
| 8     | ‘crossnorm_2’    | Cross channel normalization |
| 9     | ‘maxpool_2’      | 2x2 max pooling   |
| 10    | ‘conv_3’         | 128 7x7x128 convolutions |
| 11    | ‘relu_3’         | ReLU              |
| 12    | ‘maxpool_3’      | 2x2 max pooling   |
| 13    | ‘conv_4’         | 128 7x7x128 convolutions |
| 14    | ‘relu_4’         | ReLU              |
| 15    | ‘maxpool_4’      | 2x2 max pooling   |
| 16    | ‘conv_5’         | 128 7x7x128 convolutions |
| 17    | ‘relu_5’         | ReLU              |
| 18    | ‘maxpool_5’      | 2x2 max pooling   |
| 19    | ‘fc_1’           | 1024 fully connected layer   |
| 20    | ‘relu_6’         | ReLU              |
| 21    | ‘dropout_1’      | 50% dropout        |
| 22    | ‘fc_2’           | 1024 fully connected layer   |
| 23    | ‘dropout_2’      | 50% dropout        |
| 24    | ‘fc_3’           | 4 fully connected layer       |
| 25    | ‘dropout_3’      | 50% dropout        |
| 26    | ‘softmax’        | Softmax            |
| 27    | ‘classoutput’    | 4 classes          |
TABLE 4. Hyperparameters in the training process of CNN (Training option).

| Hyperparameters                  | Description                                      |
|----------------------------------|--------------------------------------------------|
| Plots                            | Plots to display during network training         |
| Verbose                          | Indicator to display training progress information |
| VerboseFrequency                 | Frequency of verbose printing                    |
| MaxEpochs                        | Maximum number of epochs                         |
| MiniBatchSize                    | Size of mini-batch                               |
| Shuffle                          | Option for data shuffling                         |
| ValidationData                   | Data to use for validation during training       |
| ValidationFrequency              | Frequency of network validation                   |
| ValidationPatience               | Patience of validation stopping                   |
| InitialLearnRate                 | Initial learning rate                             |
| LearnRateSchedule                | Option for dropping learning rate                |
| LearnRateDropPeriod              | Number of epochs for dropping the learning rate  |
| LearnRateDropFactor              | Factor for dropping the learning rate            |
| L2Regularization                 | Factor for $L_2$ regularization                  |
| Momentum                         | Contribution of previous step                    |
| GradientDecayFactor              | Decay rate of gradient moving average            |
| Epsilon                          | Denominator offset                               |
| ResetInputNormalization          | Option to reset input layer normalization        |
| GradientThreshold                | Gradient threshold                               |
| GradientThresholdMethod          | Gradient threshold method                         |
| SequencePaddingDirection         | Direction of padding or truncation               |
| SequencePaddingValue             | Value to pad input sequences                     |
| ExecutionEnvironment             | Hardware resource for training network           |
| WorkerLoad                       | Parallel worker load division                    |
| DispatchInBackground             | Use background dispatch                          |

TABLE 5. The hardware configuration.

| Device     | Description       |
|------------|-------------------|
| CPU        | i3 3.6 GHz        |
| GPU        | Geforce 1060 6 Gb |
| RAM        | 16 Gb             |
| HDD        | SSD 160 Gb        |

D. HYPERNET PROCESSING

Regarding the training CNN model, the task of finding and configuring parameters for training is a complicated and time-consuming process. With a certain CNN model architecture and a certain dataset, searching and selecting the training parameter set that best suits the existing model and dataset is required. Among many hyperparameter search solutions, this paper proposes using the Bayesian optimization search solution. Bayesian optimization is a suitable algorithm for optimizing the hyperparameters of classification and regression models. Bayesian optimization can be used to optimize the complex, discontinuous, and time-consuming evaluation process.

Of which, Sequential Model-Based Optimization (SMBO) is seen as a highlight hyperparameter search algorithm with its validated effectiveness [33]. Technically, the idea of this solution is that setting a representation model named M over the target black-box, then this model is updated sequentially by querying $f(\theta)$ at new locations to optimize the acquisition function throughout Expected Improvement (EI) which is calculated as follows:

$$EI_{y^*}(\theta, M) = \int_{-\infty}^{y^*} \max(y - y_0, 0) p_M(y | \theta) dy$$  \hspace{1cm} (1)$$

$y$ and $y^*$ present the real values and corresponding thresholds. The SMBO algorithm in details is shown in Algorithm 1. The operating model of the Bayesian optimization algorithm is demonstrated in Figure 3.

Algorithm 1 Sequential Model-Based Optimization SMBO($f$, $\theta$, $\Theta$, $T$)

Data: Target function $f$, Hyperparameters $\theta$ Search Space $\Theta$, Iteration $T$

Result: Best Hyperparameters $\theta^*$

Initial model $M$

$H \leftarrow \emptyset$

For $t \leftarrow 1$ to $T$

$\theta^* = \arg \max_{\theta \in \Theta} EI(\theta, M_{t-1})$;

Evaluate $f(\theta^*)$;

$H \leftarrow H \cup (\theta^*, f(\theta^*))$;

Fit model $M_t$ by $H$

Return $\theta^*$ given $H$

E. HARDWARE CONFIGURATION

To illustrate the search process of hyperparameter value and to train the PDNet model, a minimum hardware system was used while still ensuring the processing time of the system (Table 5).

V. EXPERIMENTAL RESULTS

To ensure accuracy and reliability, post-trained PDNet-Vehicle and PDNet-TrafficSign models were evaluated on an independent test dataset that had never been used before. Each test dataset contained data that included possible cases of actual vehicles and traffic signs. To retrain PDNet models, the use of the OptionTrain training parameter set corresponding to each type of object is suggested: OptionTrain_Vehicle and OptionTrain_TrafficSign.

Among the hyperparameters in Table 4, the selection of only six important parameters directly affecting the accuracy of the model and training time was proposed. Selecting too many hyperparameters would not provide a real idea of the process due to the amount of time consumed if the Bayesian algorithm were expected to work optimally. In contrast to the limited time and large number of hyperparameters to be searched, the Bayesian algorithm would fail to choose a truly
optimal hyperparametric solution. In addition, the search domain of six hyperparameters was also considered and adjusted appropriately, avoiding the wide parameter domain that affects accuracy and search time. From the experimental process, the proposed parameter value domain of six hyperparameters is shown in Table 6.

**A. TRAINING THE INITIAL PDNET MODEL**

The initial PDNet architecture (Table 3) was initialized and PDNet-Vehicle₀ and PDNet-TrafficSign₀ were trained using the PDNet-Vehicle data and PDNet-TrafficSign data correspondence, as shown in Table 7 and Table 8, respectively. The unified dataset was used for evaluation on both the optimal parameters using Bayesian algorithms and accuracy testing of the trained PDNet model. The accuracy of the initial PDNet-Vehicle and PDNet-TrafficSign models is shown in Figure 4.

Trained PDNet-Vehicle and PDNet-TrafficSign models were named PDNet-Vehicle₀ and PDNet-TrafficSign₀.

**FIGURE 4.** The confusion matrix of the accuracy of initial PDNet-Vehicle and PDNet-TrafficSign model.

**TABLE 6.** Model parameter domain values.

| Hyperparameters | Values range |
|-----------------|--------------|
| InitialLearnRate | [0.0001 0.9] |
| Momentum        | [0.1 0.99]   |
| L2Regularization| [0.0001 0.01]|
| MiniBatchSize   | [64 256]     |
| GradientThreshold| [10 10²]   |
| GradientThresholdMethod | [‘2norm’, ‘global-2norm’, ‘absolute-value’] |

**TABLE 7.** Image datasets for training initial PDNet-Vehicle.

| Dataset name | Quantity |
|--------------|----------|
| Motorcycle   | 600      |
| Car          | 600      |
| Coach        | 600      |
| Truck        | 600      |
| Negative     | 300      |

**TABLE 8.** Image datasets for training initial PDNet-TrafficSign.

| Dataset name | Quantity |
|--------------|----------|
| No Entry     | 290      |
| No parking, stopping | 209      |
| Non-priority intersection | 199      |
| Negative     | 306      |

**TABLE 9.** Image datasets for testing PDNet-Vehicle model Image data (Data-Vehicle₀) for searching hyperparameters and the PDNet-Vehicle₁ model.

| Class              | Quantity of confident tracking | Quantity of last model (70%) | Total data for retraining |
|--------------------|--------------------------------|------------------------------|---------------------------|
| Motorcycle         | 186                            | 420                          | 606                       |
| Car                | 195                            | 420                          | 615                       |
| Coach              | 190                            | 420                          | 610                       |
| Truck              | 200                            | 420                          | 620                       |
| Negative           | 90                             | 210                          | 300                       |

**TABLE 10.** Image datasets for testing PDNet-TrafficSign model Image data (Data-TRAFFICSIGN₀) for searching hyperparameters and the PDNet-TRAFFICSIGN₁ model.

| Class              | Quantity of confident tracking | Quantity of last model (70%) | Total data for retraining |
|--------------------|--------------------------------|------------------------------|---------------------------|
| No Entry           | 91                             | 263                          | 294                       |
| Noparking, stopping| 70                             | 147                          | 217                       |
| Non-priority intersection | 60                         | 140                          | 200                       |
| Negative           | 108                            | 210                          | 318                       |

**B. OPTIMIZATION OF LEARNING PARAMETERS, UPDATE PDNET MODEL**

During ADAS movement, the IONet model and PDNet model were used by the system to recognize and acquire continuously new image data from objects. Once the number of images reached N value, the system would initiate a pre-retrain dataset with X% of the dataset coming from the newly acquired image and [100-X] % from the previous training dataset. Using an old data part was aimed at avoiding overfitting. From the experiment, a value at X = 30% of the number of images of the previous smallest dataset was proposed.

For PDNet-Vehicle: N = 30% × 277 = 90 images.
For PDNet-TrafficSign: N = 30% × 277 = 90 images.

Data for retraining PDNet-Vehicle₀ (referred to as Data-Vehicle₀) and PDNet-TrafficSign₀ (referred to as Data-TRAFFICSIGN₀) are shown in Tables 9 and 10.

Data-Vehicle₀ and Data-TRAFFICSIGN₀ are the new dataset. Thus, hyperparameters (OptionReTrain_Vehicle₀ and OptionReTrain_TrafficSign₀) needed to be searched for PDNet-Vehicle₀ and PDNet-TrafficSign₀ models that were fixed with these new ones.
The Bayesian algorithm was used to search for hyperparameters with the architecture of the fixed CNN model (PDNet model, Table 3). In this study, we investigated $\text{MaxObjectiveEvaluations} = 60$, $\text{MaxEpochs} = 200$ and the value domain of hyperparameters need to be found, as shown in Table 6. Based on the experimental process, it was decided that $\text{MaxEpochs} = 200$ was appropriate to ensure the training time and the accuracy of the model. The model found by the Bayesian algorithm was simply the trained model on the optimal parameters. Therefore, it was not necessary to use hyperparameters to retrain PDNet models. Figure 5 displays the Bayesian function’s objective value evaluated on objective function evaluations. The confusion matrix for the test data in the search process of hyperparameter values is shown in Figure 6 (PDNet-TrafficSign). Table 11 presents the results of searching for the hyperparameter values of the PDNet-Vehicle and the PDNet-TrafficSign models.

The Bayesian algorithm was applied to estimate the optimal hyperparameters for training PDNet-Vehicle$_1$ and PDNet-TrafficSign$_1$ models using PDNet-Vehicle$_0$ and PDNet-TrafficSign$_0$. The confusion matrix of the accuracy of PDNet-Vehicle$_1$ and PDNet-TrafficSign$_1$ model is shown in Figure 7, with an accuracy of 62.3% and 85.2%, respectively. It can be seen from the evaluation results on the same dataset that the recognition accuracy of PDNet-Vehicle$_1$ and PDNet-TrafficSign$_1$ was higher than those of PDNet-Vehicle$_0$, PDNet-TrafficSign$_0$. Thus, these models were in turn updated, and they replaced the old recognition model in the ADAS.

The ADAS continued to recognize and acquire new data of objects over time. The training dataset was created as the number of images reached the N value. The dataset for retraining PDNet-Vehicle$_1$ (Data-Vehicle$_1$) and PDNet-TrafficSign$_1$ (Data-TrafficSign$_1$) models is shown in Tables 12 and 13.

It continued to search the hyperparameters (OptionReTrain _Vehicle$_1$ and OptionReTrain_TrafficSign$_1$) that matched the newly found datasets for PDNet-Vehicle$_1$ and PDNet-TrafficSign$_1$. The search result of the optimal hyperparameter values and model using Bayesian algorithm is shown in Table 14.
FIGURE 9. Comparing the accuracy of recognition results of Vehicle and Traffic sign models.

![Figure 9](image1)

FIGURE 10. The confusion matrix for the accuracy of AlexNet model for vehicle recognition.

![Figure 10](image2)

FIGURE 11. The confusion matrix for the accuracy of AlexNet model for traffic sign recognition.
The PDNet-Vehicle\textsubscript{1} and PDNet-TrafficSign\textsubscript{1} models that were searched (with the Bayesian algorithm) were referred to as PDNet-Vehicle\textsubscript{2} and PDNet-TrafficSign\textsubscript{2}. The confusion matrix of the accuracy of the PDNet-Vehicle\textsubscript{2} and PDNet-TrafficSign\textsubscript{2} model is shown in Figure 8, with an accuracy of 70.0% and 92.9%, respectively.

The intelligence of ADAS was constantly improved without any human intervention due to its continuous operation during the on-the-road journey. There were changes in model accuracy when comparing the vehicle and traffic sign recognition results of the initial model (PDNet-Vehicle\textsubscript{0} and PDNet-TrafficSign\textsubscript{0}) and optimal models (PDNet-Vehicle\textsubscript{1} and PDNet-TrafficSign\textsubscript{1}) shown in Figure 9.

C. COMPARING WITH THE STATE-OF-THE-ART MODELS
Based on the results obtained through the adaptive learning emulation of PDNet-Vehicle and PDNet-TrafficSign models,
Hyperparameter Optimization for Improving Recognition Efficiency of an Adaptive Learning System

FIGURE 15. The chart showing the increasing accuracy on recognition of the Vgg model after updating the recognition model using optimal hyper parameters.

TABLE 14. Found optimal hyperparameter values of PDNet-Vehicle$_2$ and PDNet-TrafficSign$_2$ model.

| Hyperparameters          | OptionReTrain Vehicle$_1$ | OptionReTrain TrafficSign$_1$ |
|--------------------------|----------------------------|------------------------------|
| InitiaLearnRate          | 0.0030                     | 1.1968e-04                   |
| Momentum                 | 0.9115                     | 9747                         |
| L2Regularization         | 0.0034                     | 0.0068                       |
| MiniBatchSize            | 247                        | 79                           |
| GradientThreshold        | 621                        | 5.2430e+12                   |
| GradientThresholdMethod  | Absolute-value             | Absolute-value               |

TABLE 15. Compared results to the previous contribution ([1]).

| Models                     | Our method (%) | Previous method ([1]) (%) |
|----------------------------|----------------|----------------------------|
| PDNet-Vehicle$_1$ (initial model) | 51.77           | 51.77                       |
| PDNet-Vehicle$_1$          | 62.30           | 60.58                       |
| PDNet-Vehicle$_2$          | 69.98           | 68.41                       |
| PDNet-TrafficSign$_1$ (initial model) | 70.46           | 70.46                       |
| PDNet-TrafficSign$_1$      | 85.19           | 84.93                       |
| PDNet-TrafficSign$_2$      | 92.90           | 90.36                       |
| AlexNet-Vehicle$_2$ (initial model) | 66.14           | 66.14                       |
| AlexNet-Vehicle$_1$        | 88.24           | 86.61                       |
| AlexNet-Vehicle$_2$        | 90.75           | 90.40                       |
| AlexNet-TrafficSign$_1$ (initial model) | 67.05           | 67.05                       |
| AlexNet-TrafficSign$_1$    | 88.78           | 87.73                       |
| AlexNet-TrafficSign$_2$    | 93.51           | 92.55                       |
| Vgg-Vehicle$_1$ (initial model) | 71.46           | 71.46                       |
| Vgg-Vehicle$_1$            | 93.11           | 92.42                       |
| Vgg-Vehicle$_2$            | 94.78           | 94.14                       |
| Vgg-TrafficSign$_1$ (initial model) | 70.46           | 70.46                       |
| Vgg-TrafficSign$_1$        | 95.27           | 94.74                       |
| Vgg-TrafficSign$_2$        | 95.53           | 94.74                       |

VI. CONCLUSION

The research content and proposal emulated the operation of ADAS in practice. Even though testing was made on only two objects (vehicle and traffic signs), they were representative and covered all possible objects of the on-the-road journey of ADAS. Moreover, the proposed model is expected to be widely applied in all intelligent systems using object recognition complexes. The results of the proposed method have provided several contributions:

1) It demonstrated that adaptive learning methods were effective, improving performance and diversifying the recognition mode of an intelligent system without relying on any human intervention. In particular, the system had the capacity to continuously learn and be ‘smart’ during its operation.

2) It supports to improve training and adaptive parameters on each dataset and created a rather comprehensive proposed model in terms of adaptive learning in intelligent systems.

3) The proposed model matched with systems with low equipment configuration, thus lacking resources for complex or multiple object recognition. Throughout the adaptive learning process of the proposed model, the system was able to recognize objects in high accuracy. The method is equivalence or higher over time other approach.
However, the following steps need to be taken to make the proposed solution possible and to improve recognition performance:

1) The recognized objects should be expanded to diversify the capabilities of the ADAS system or to develop it into a complete robotic system capable of adaptive learning for all subjects.

2) The number and value domain of the hyperparameters adapting to new datasets should be expanded before training the recognition models.

Generally, even though the proposed solution does not yet cover all recognition cases of ADAS, it offers a quite comprehensive model for adaptive learning in the future.

REFERENCES

[1] D.-P. Tran and V.-D. Hoang, “Adaptive learning based on tracking and reidentifying objects using convolutional neural network,” *Neural Process. Lett.*, vol. 50, no. 1, pp. 263–282, Aug. 2019.

[2] G. Li, S. E. Li, R. Zou, Y. Liao, and B. Cheng, “Detection of road traffic participants using cost-effective arrayed ultrasonic sensors in low-speed traffic situations,” *Mach. Syst. Signal Process.*, vol. 132, pp. 535–545, Oct. 2019.

[3] S. E. Li, G. Li, J. Yu, C. Liu, B. Cheng, J. Wang, and K. Li, “Kalman filter-based tracking of moving objects using linear ultrasonic sensor array for road vehicles,” *Mach. Syst. Signal Process.*, vol. 98, pp. 173–189, Jan. 2018.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[6] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[8] R. Girshick, “Fast R-CNN,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[9] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[10] G. Li, Y. Yang, and X. Qu, “Deep learning approaches on pedestrian detection in hazy weather,” *IEEE Trans. Ind. Electron.*, vol. 67, no. 10, pp. 8889–8899, Oct. 2020.

[11] D.-P. Tran and V.-D. Hoang, “Vehicle categorial recognition for traffic monitoring in intelligent transportation systems,” in *Proc. Asian Conf. Intell. Inf. Database Syst.*, 2019, pp. 670–679.

[12] D.-P. Tran, N. G. Nhu, and V.-D. Hoang, “Pedestrian action prediction based on deep features extraction of human posture and traffic scene,” in *Proc. Asian Conf. Intell. Inf. Database Syst.*, 2018, pp. 563–572.

[13] K. Blix and T. Eiolf, “Machine learning automatic model selection algorithm for oceanic chlorophyll—A content retrieval,” *Remote Sens.*, vol. 10, no. 5, p. 775, May 2018.

[14] S. Raschka, “Model evaluation, model selection, and algorithm selection in machine learning,” 2018, arXiv:1811.12808. [Online]. Available: http://arxiv.org/abs/1811.12808

[15] L. Li and A. Talwalkar, “Random search and reproducibility for neural architecture search,” 2019, arXiv:1902.07638. [Online]. Available: http://arxiv.org/abs/1902.07638

[16] H. Bertrand, R. Ardon, M. Perrot, and I. Bloch, “Hyperparameter optimization of deep neural networks: Combining hyperband with Bayesian model selection,” in *Proc. Conférence sur l’Apprentissage Automatique*, 2017, pp. 1–5.

[17] T. Domhan, J. T. Springenberg, and F. Hutter, “Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves,” in *Proc. 24th Int. Joint Conf. Artif. Intell.*, 2015, pp. 1–9.
GIA-NHU NGUYEN (Member, IEEE) received the Ph.D. degree in mathematical for computer science from the Ha Noi University of Science, Vietnam National University, Vietnam. He is currently the Dean of Graduate School with Duy Tan University, Vietnam, where he is also a Professor. He has a total academic teaching experience of 19 years with more than 60 publications in reputed international conferences, journals, and online book chapter contributions (Indexed By: SCI, SCIE, SSCI, Scopus, ACM DL, and DBLP). His areas of research interests include healthcare informatics, network performance analysis and simulation, as well as computational intelligence.

VAN-DUNG HOANG received the Ph.D. degree from the University of Ulsan, South Korea, in 2015. He has been serving as a Professor in computer science with AI Laboratory, Faculty of Information Technology, Ho Chi Minh City University of Technology and Education, Vietnam. He has published numerous research articles in ISI, Scopus indexed, and high-impact factor journals. His research interests include a wide area, which focuses on pattern recognition, machine learning, medical image processing, computer vision application, vision-based robotics and ambient intelligence, and communication networks.