A Fast Military Object Recognition using Extreme Learning Approach on CNN

Hari Surrisyad[1]
Master Program in Computer Science
Universitas Gadjah Mada, Yogyakarta, Indonesia

Wahyono[2]*
Department of Computer Science and Electronics
Gadjah Mada University, Indonesia

***Abstract*—Convolutional Neural Network (CNN) is an algorithm that can classify image data with very high accuracy but requires a long training time so that the required resources are quite large. One of the causes of the long training time is the existence of a backpropagation-based classification layer, which uses a slow gradient-based algorithm to perform learning, and all parameters on the network are determined iteratively. This paper proposes a combination of CNN and Extreme Learning Machine (ELM) to overcome these problems. Combination process is carried out using a convolution extraction layer on CNN, which then combines it with the classification layer using the ELM method. ELM method is Single Hidden Layer Feedforward Neural Networks (SLFNs) which was created to overcome traditional CNN's weaknesses, especially in terms of training speed of feedforward neural networks. The combination of CNN and ELM is expected to produce a model that has a faster training time, so that its resource usage can be smaller, but maintaining the accuracy as much as standard CNN. In the experiment, the military object classification problem was implemented, and it achieves smaller resources as much as 400 MB on GPU comparing to standard CNN.**

***Keywords*—Training-speed; resource; backpropagationm; CNN; ELM**

I. Introduction

In recent years, the field of computer vision has been developed to support advanced systems in various fields such as intelligent robots, automatic control systems, and human-computer interaction. On the other hand, one of the applications in the military field is automatic target detection, which is the main technology for automatic military operations and surveillance missions [1]. Military objects are legitimate targets for attack in war [2].

Convolutional Neural Network (CNN) is a popular algorithm that excels in vector data classification, which belongs to deep learning algorithms. CNN is a special type of neural network that handles phenomena such as localization of receptive fields in large data volumes, copying weights forward, as well as image sampling using different kernels in each convolution layer [3]. Convolution is a process in which an image is manipulated by using an external mask to produce a new image [4]. CNN uses a feedforward neural network with backpropagation-based learning at its classification layer or what is often called the fully connected layer. The feedforward neural network has the disadvantage of using a slow gradient-based learning algorithm for learning [5]. All parameters on the feedforward neural network must be determined manually iteratively, the parameters in question are the input weight and hidden bias. These parameters are also interconnected between layers, so they are often stuck on the local optima and require a long learning time and lots of resources.

Extreme Learning Machine (ELM) is a feedforward neural network with a single hidden layer or commonly known as Single Hidden Layer Feedforward Neural Networks (SLFNs). The ELM learning method can overcome weaknesses of CNN, especially in terms of rapid training of the feedforward neural network [5]. Therefore, a combination of convolutional neural networks and extreme machine learning is proposed, by replacing the backpropagation method used at the CNN classification layer with the ELM method which can overcome the weakness of backpropagation. This combination is expected to increase learning speed become faster so that the utilization of resources during training is getting smaller, but with accuracy the same as for regular CNN.

This research is expected to be used in situations where a system, especially in the military field, requires small resources and prioritizes speed. An example of a real implementation that can be done is in a surveillance drone, which can recognize military objects. Therefore, drones and adjust the distance to the recognized objects. In this case, the military objects can be recognized from far distance, such as military aircraft, military helicopters, and others, as well as at close range such as grenades, pistols, rifles, and so on.

II. Related Work

Many researches related to the introduction of military objects with CNN and ELM have been carried out. One of them is deep transfer learning for military object Recognition under small training set condition [6]. This research focuses on the classification and recognition of objects with a limited amount of data with CNN, with transfer learning to provide knowledge and combining various layers to perform better feature extraction. It obtained an average value of 95% accuracy. Another method is recognizing military vehicles in social media images using deep learning [7]. The research was evaluated using dataset which was collected from various social media, namely Flickr, YouTube, and the Web. In the experiment, it achieved an accuracy of 95.18%. However, both method still requires slow processing time and large resource in training step.

ELM is a feedforward neural network with a single hidden layer or commonly called a single hidden layer feed-forward

*Corresponding Author

neural network, which has advantages in learning speed. One of the ELM utilization is for recognizing facial expression which was done by Mahmud and Al Mamun [8]. In this research, facial expression image recognition was classified into six classes, with ELM and Backpropagation Neural Network as a comparison, ELM obtained an accuracy of 90% and backpropagation 86%, while the ELM speed was 0.0936 second and backpropagation 1 second with a total of 42 image data. Another utilization of ELM is proposed by Wiyono which implemented ELM as classifier for face recognition combining with PCA for feature extraction [9]. By using JAFFE Dataset, the method obtained an accuracy of 93.1% with a training speed of 0.062 seconds.

Research combining CNN with other algorithms is also nothing new, one of them is Convolutional SVM Networks for Object Detection in UAV Imagery proposed by Bazi and Melgani [10]. In this research, the network used is based on several alternatives convolutional and a reduction layer, which was then combined with the SVM as classification layer. This is done in order to obtain an optimal model for classification and prediction, with very limited training data. The resulting accuracy in this research is 97% for the Car dataset and 96% for the solar panel dataset.

III. Methodology

A. Research Goal

In this research, we aim to overcome the weaknesses of backpropagation used in convolutional neural networks. It is expected that the proposed method could increase the speed of training step so that the resources used are also getting smaller. The flow of our proposed method is shown in Fig. 1.

B. Data Acquisition

In this research, we collected military object image data which consists of 16 different classes, with 15 military object classes and 1 non-military object class. The data was collected from Google images, using Google-images-download library, this library is made with the Python programming language. It is then divided into training and testing data, along with the object image class to be used. Fig. 2 shows several selected samples of military object images collected in our dataset.

C. Data Preprocessing

Data preprocessing is a series of processes carried out on data so that the data is ready to be used as input in the training process. There are many types of image data preprocessing that can be done. In this research, the preprocessing that will be carried out is as follows:

1) Data cleaning: After the data is obtained during the acquisition process, the data will be cleaned first. The cleaning process is carried out by deleting data that does not match the criteria as follows: (1) data in the form of weapons that are not being held or used, (2) vehicle data, taken from the side or tilt angle that represents the shape of the vehicle in general, (3) data according to the object class that has been defined, and (4) the image data only containing one object, except the army object class.

2) Data augmentation: The next preprocessing is data augmentation. The data augmentation is required because the number of data obtained in the dataset is very limited, such as only 350 data per class. For obtaining a good classification accuracy, the data should be large enough. However, it would be very difficult to collect such large data manually, so that we employ the data augmentation for increasing the number of data. Data augmentation example is shown in Fig. 3.

3) Resizing: The next step is resizing the image data. This resizing process is carried out to equalize the image size of each data, because the data obtained from Google images have various sizes and dimensions. In this research, image data will be resized to 224 × 224 pixels similar to our input layer size on the CNN architecture. The illustration of resizing process is depicted in Fig. 4.

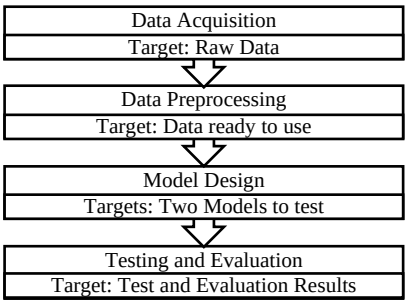

Fig. 1. Research Flow.

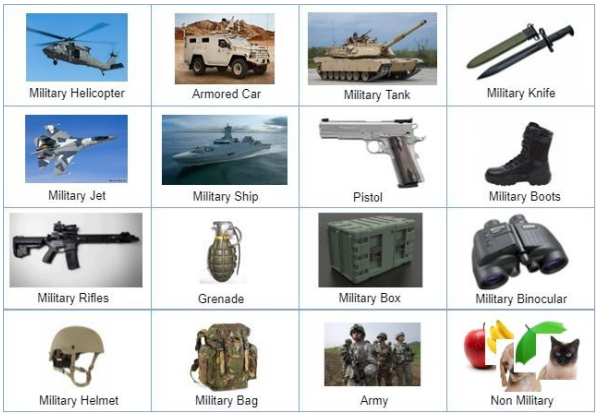

Fig. 2. Sample of Military Object Images used in our Proposed Dataset.

(a) (b)

Fig. 3. Data Augmentation Ilustration using Horizontal Flip (a) Raw Data (b) Augmentation Results.

(a) (b)

Fig. 4. Resize the Image to 224 × 224.

D. Model Design

In this research, after the data is ready, a model design process will be carried out to perform the learning process of the training data from each class. The designed model will greatly affect the classification results.

1) Normal CNN: CNN is a convolutional operation that combines multiple layers of processing, uses several elements operating in parallel and is inspired by the biological nervous system [11]. In CNN, each neuron is represented in two dimensions, so this method is suitable for processing with input in the form of images [12]. The CNN structure consists of input, feature extraction process, classification process and output. The extraction process on CNN consists of several hidden layers, namely the convolution layer, the activation function (ReLU), and pooling, as shown in Fig. 5.

In designing the CNN model, there are many types of architectures that can be made. Each architecture with certain data must go through a tuning process to get a model that is considered optimal. Fig. 6 is the initial architecture that will be used for further tuning in this research.

The tuning process is carried out by making gradual changes to the initial architecture that has been determined. There are many parameters that can be set in the CNN model such as number of convolution and concatenation operations, order of each operation, kernel in convolution and concatenation operations, number of hidden layers in FCL, number of nodes in each hidden layer and many more. Tuning process will be stopped if the optimal model has been found.

2) Combination of CNN and ELM: ELM is a feedforward neural network with a single hidden layer or commonly called Single Hidden Layer Feedforward Neural Networks (SLFNs) which only requires two parameters, namely the number of hidden nodes and the choice of activation function. The ELM learning method is designed to overcome the weaknesses of the feedforward neural network, especially in terms of learning speed. Based on two reasons why feedforward ANN has a slow learning speed:

- Using slow gradient based learning algorithms for conducting training.
- All parameters on the network are determined iteratively using this learning method.

In ELM, parameters such as input weights and hidden bias are chosen randomly, so that ELM has the ability to learn quickly and is able to produce good generalization performance.

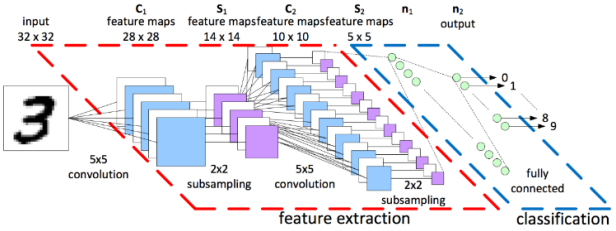

Fig. 5. CNN Architecture Baseline [13].

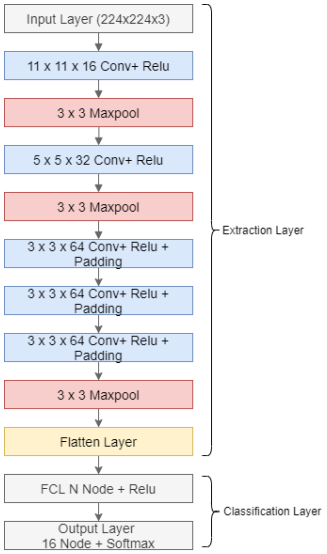

Fig. 6. Initials CNN Architecture.

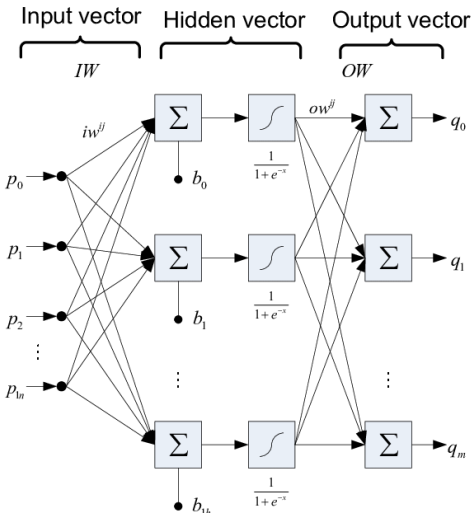

Fig. 7. ELM Network [14].

The ELM method has a different mathematical model from the feedforward neural network, as shown in Fig. 7. The ELM mathematical model is simpler and more effective. For N different number of input pairs and output targets (x_i, t_i), with $x_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^T \in \boldsymbol{R}^n$ and $t_i = [t_{i1}, t_{i2}, \ldots, t_{in}]T \in \boldsymbol{R}^m$, Standard SLFN with the number of hidden nodes and the activation function $g(x)$ can be modeled mathematically as follows:

$$\sum_{i=1}^{\tilde{N}} \beta_i g_i(x_j) = \sum_{i=1}^{\tilde{N}} \beta_i g(w_i \cdot x_j + b_i) = o_j, j = 1, 2, \ldots, N \quad (1)$$

where:

$w_i = [w_{i1}, w_{i2}, \ldots, w_{in}]^T$ is a weight vector that connects *hidden node* i and *input nodes*.

$\beta_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{im}]^T$ is the connecting weight vector *hidden node* i and *output nodes*.

bi is threshold from hidden node i

$w_i \cdot x_j$ is *inner product* from w_i and x_j

Standard SLFNs with $\tilde{N}$ hidden nodes and activation function $g(x)$ assumed to be able to estimate N of this sample with an error rate of 0 which means $\sum_{j=1}^{N}\|o_j - t_j\| = 0$, so there is β_i, w_i, and b_i that:

$$\sum_{i=1}^{\tilde{N}} \beta_i g(w_i . x_j + b_i) = t_j , j = 1,2, \dots, N \quad (2)$$

The above equation can be simply written as:

$$H\beta = T \quad (3)$$

where:

$$H = \begin{bmatrix} g(w_1 . x_1 + b_1) & \cdots & g(w_{\tilde{N}} . x_1 + b_{\tilde{N}}) \\ \vdots & \ddots & \vdots \\ g(w_1 . x_n + b_1) & \cdots & g(w_{\tilde{N}} . x_N + b_{\tilde{N}}) \end{bmatrix},$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}^T \end{bmatrix} and\ T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}$$

H in the above equation is the hidden layer output matrix of the neural network. $g(w_i . x_j + b_i)$ shows the output of hidden neurons related to input xj. β is the output weight matrix and T is the target matrix. In ELM, the input weight and hidden bias are determined randomly, so that the output weight associated with the hidden layer can be determined from the equation:

$$\beta = H^+ T \quad (4)$$

In the equation above H^+ is the *Moore-Penrose Generalized invers* matrix of the H matrix. H^+ is obtained by the equation:

$$H^+ = (H^T .H)^{-1}.H^T \quad (5)$$

H is the hidden layer output matrix and H^T is the transpose of H. Following are the steps in the Extreme Learning Machine (ELM) algorithm:

Input : input pattern x_j and target output pattern t_j, j = 1, 2,..N

Output: input weight w_i, output weight β_i and *bias* b_i, i = 1,2...$\tilde{N}$

Steps :

1: Determine the activation function ($g(x)$) and the number of hidden nodes ($\tilde{N}$).

2: Determine the random value of the input weight w_i and *bias* b_i, i = 1, 2, ..., $\tilde{N}$.

3: Calculate the output matrix value H on the hidden layer.

4: Calculate the output weight value β using $\beta = H^+ T$.

5: Calculate the output value with $H\beta = T$.

In this research, the combination layer feature extraction model of CNN and ELM will use the same layer as the feature extraction layer in the normal CNN model that has been tuned. The difference is that this combination model classification layer will replace the FCL which uses backpropagation as the basis for learning with ELM.

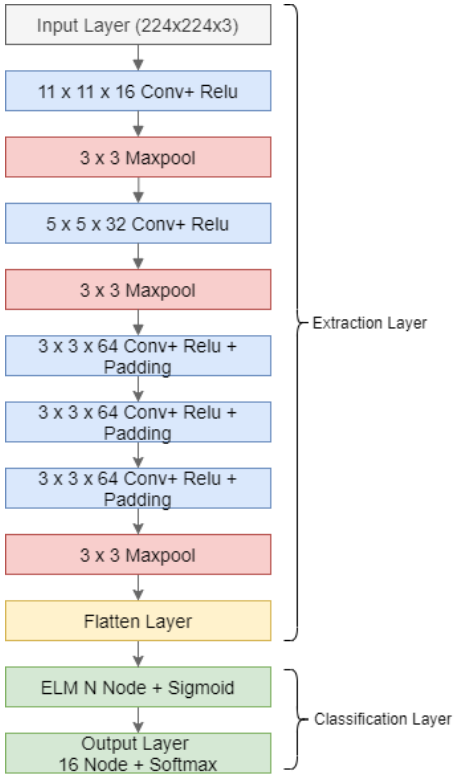

Fig. 8. Initial Combined Architecture of CNN and ELM.

The ELM classification layer will be tuned again. However, the only parameters that will be tuned are the number of nodes in the hidden layer and their activation function. On the other hand, the number of hidden layers will not be set because basically ELM is a single hidden layer feedforward neural network (SLFNs), as shown in Fig. 8.

E. Testing and Evaluation Design

After the model design process is complete, two different models will be obtained, namely normal CNN and Combination of CNN and ELM. Furthermore, several testing and evaluation processes will be carried out. Fig. 9 is the test and evaluation design scheme that will be carried out in this research.

Fig. 9. Testing and Evaluation Design.

1) Testing speed of training and resource usage: In the testing process, the two methods will be implemented and then calculated how long it will take for the training time, this measurement will be done in seconds. Testing the use of resources required by both methods, the resource referred to here is the use of memory during the training process.

Testing of training speed and resource usage is conducted on several factors, such as, the amount of data, the variation of the extraction layer, the number of hidden layers (FCL classification layer), and the number of hidden layer nodes (classification layer). The detail comparison schema for training speed and resources is shown in Table I.

TABLE I. TRAINING SPEED TESTING SCHEME AND RESOURCE USAGE

Model	Factor	Testing Speed	Testing Resources
Normal CNN	The amount of data	√	√
	Extraction layer variations	√	√
	Number of hidden layers (classification layer)	√	√
	Number of hidden layer nodes (classification layer)	√	√
Proposed Combination of CNN and ELM	The amount of data	√	√
	Extraction layer variations	√	√
	Number of hidden layers (classification layer)	×	×
	Number of hidden layer nodes (classification layer)	√	√

2) Cross validation evaluation: Furthermore, the cross-validation evaluation process will be carried out on the two models that have been made. Cross-validation was carried out to evaluate the accuracy of the two models that have been made against the training data.

All training data will be divided into n subsets evenly with the same size, then the training and testing process is carried out n times repeatedly. In iteration 1, subset 1 becomes validation data and the other becomes training data. In iteration 2, subsets 2 becomes validation data and others become training data, and so on until it has been finished, as shown in Fig. 10.

3) Accuracy, precision, and recall evaluation: The final evaluation that will be carried out is the evaluation process on the test data. This process is applied by classifying all test data which also has 15 classes. These data are data that are not included in the training process. Then the process of calculating accuracy, precision and recall will be carried out using following equations:

$$Accuracy = \frac{\sum_{i=1}^{l} \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}}{l} * 100\% \quad (6)$$

$$Precision_{class} = \frac{TP_{class}}{TP_{class} + FP_{class}} * 100\% \quad (7)$$

$$Recall_{class} = \frac{TP_{class}}{TP_{class} + FN_{class}} * 100\% \quad (8)$$

where

- TP_i is *True Positive*, that is, the number of positive data classified correctly by the system for class i.
- TN_i is *True Negative*, that is, the number of negative data classified correctly by the system for class i.
- FN_i is *False Negative*, that is, the amount of negative data but incorrectly classified by the system for class i.
- FP_i is *False Positive*, that is, the number of positive data but incorrectly classified by the system for class i.
- l is number of classes.

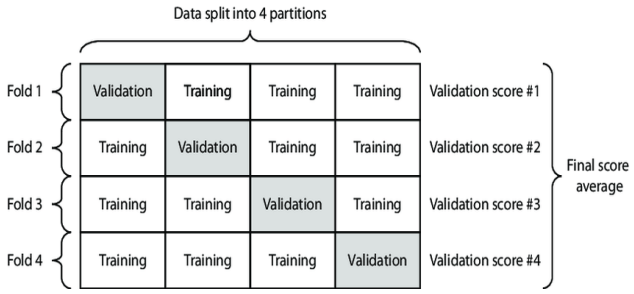

Fig. 10. Illustration of 4-Fold Cross Validation.

IV. EXPERIMENTS RESULTS

A. Data Acquisition Results

The data acquisition process that has been carried out using the *google_images_download* library with various keywords in each object class, has succeeded in collecting 16 classes of raw data with different amounts of data in each class. The results of data acquisition can be seen in the Fig. 11.

B. Results of Data Preprocessing

The raw data that has been collected will then go through several preprocessing stages, this is done to compile the raw data into data that is ready for use, and several processes are carried out as follows:

1) Cleaning data: The first step is the cleaning process. This process is carried out to clean data that is incompatible with existing classes. The result of this process is data that contains and is in accordance with the existing class, in each class 350 images are selected, so that the total data in the data set is 5,600 images.

2) Resizing: The next process is resizing data. All the data that have been selected have a very diverse size. To simplify the modeling process, all data will be equalized in pixel size to 224 × 224 pixels. An example of the results of the resizing process can be seen in Fig. 12.

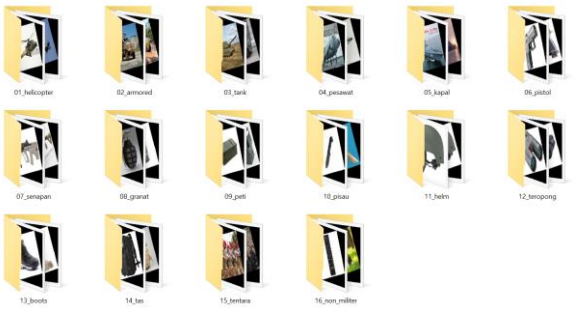

Fig. 11. Data Acquisition Results.

Fig. 12. Resizing Data Results.

3) Augmentation data: The next process is data augmentation. This process is conducted to increase the amount of data, so that the model has enough data. Therefore, it can be used for the training process and produces a good model. Some of the data augmentation used are as follows:

- Flip Horizontal

The first augmentation is a horizontal flip. This process is performed to flip the image horizontally. In the result of this process, the data is duplicated, so that each class will have a total of 700 data, and the total data in the dataset is 11,200. An example of the results of the flipping process can be seen in the Fig. 13.

- Rotating

The second augmentation is rotating. This process is applied to rotate the image, in this research the image is rotated. The result of this process the data is increased threefold, each class the number becomes 1,050 data, so that the total data in the dataset is 16,800a. This data will be used in the modeling process. An example of the result of the rotation process can be seen in the Fig. 14.

- Shifting

The third augmentation is shifting. This process is performed to shift the position of the pixels in the image. In this research the pixels are shifted 30 pixels to the right. The result of this process the data is increased fourfold, each class the number becomes 1,400 data so that the total data is 22,400 data. This data will be used in the testing process. An example of the results of the shifting process can be seen in the Fig. 15.

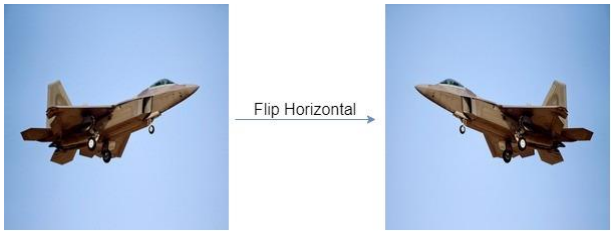

Fig. 13. Results of the Flipping Process.

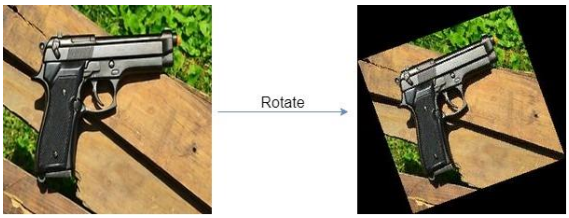

Fig. 14. Results of the Rotation Process.

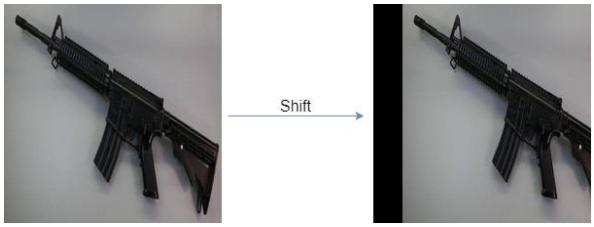

Fig. 15. Result of the Shift Process.

C. Modeling Results

Data that is ready and has a sufficient amount will be used in the modeling process. The data used in the modeling process is 1,050 per class and a total of 16,800 data as a whole. The data will be divided into training and testing data with a ratio of 80:20. After conducting experiment, the modeling results are as follows.

1) Normal CNN model: The first modeling process is Normal CNN, with the initial architecture that has been determined at the beginning of the research. The results of the training are as shown in Fig. 16.

In the training process above, the training time is 2 minutes 49 seconds, with peak resource usage of 123.8% CPU, 3032 MB RAM, and 293 MB GPU. In the training process, it obtains accuracy of 0.987, while the data test was 0.890.

From the initial architecture, the tuning process is conducted. After going through a long, we obtain the optimal architecture as shown in Fig. 17.

Fig. 18 shows the results of training from tuned CNN architectures. In the training process with a tuned architecture, the training time is 4 minutes 30 seconds, with peak resource usage, such as CPU 156.8%, RAM 3300 MB, and GPU 771 MB. It obtains the training accuracy of 0.984 while the test data was 0.924.

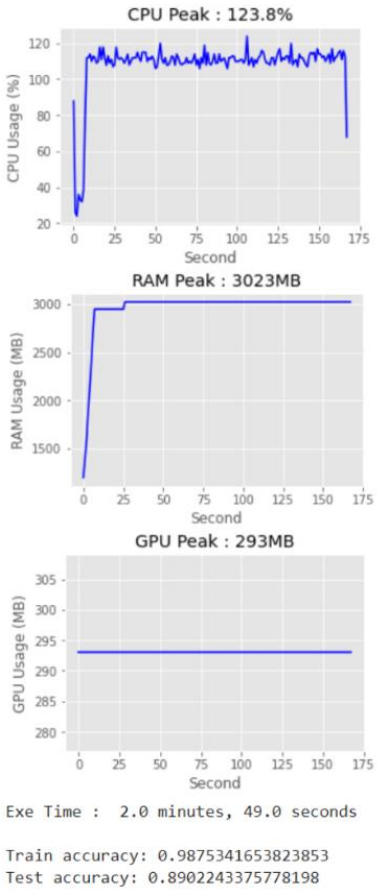

Fig. 16. Initial Normal CNN Architecture Training Results.

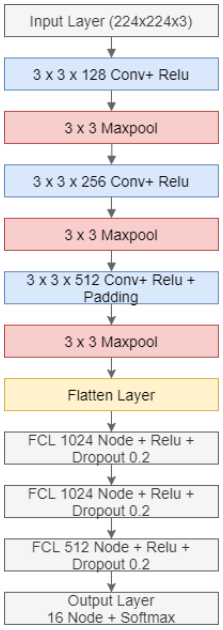

Fig. 17. Optimal CNN Architecture Obtaining by Tuning Process.

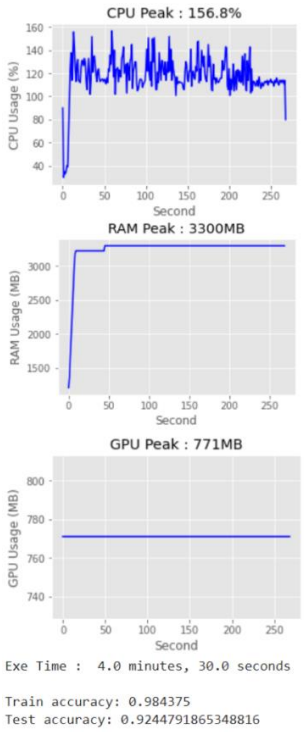

Fig. 18. The Tuned CNN Normal Architecture Training Results.

2) Combination of CNN and ELM model: The next modeling process is the combination modeling of CNN and ELM. Using the initial architecture, the training results are shown in Fig. 19.

In the training process of the combined CNN and ELM model with the initial architecture, the training speed is 52 seconds, with peak resource usage of 197.9% CPU, 4327 MB RAM, and 229 MB GPU. The accuracy in training was 0.903 while the test data is 0.815.

The combined CNN and ELM model also goes through a tuning process, and the tuning results are shown in Fig. 20. Fig. 21 shows the results of training from Combined Architecture of CNN and ELM.

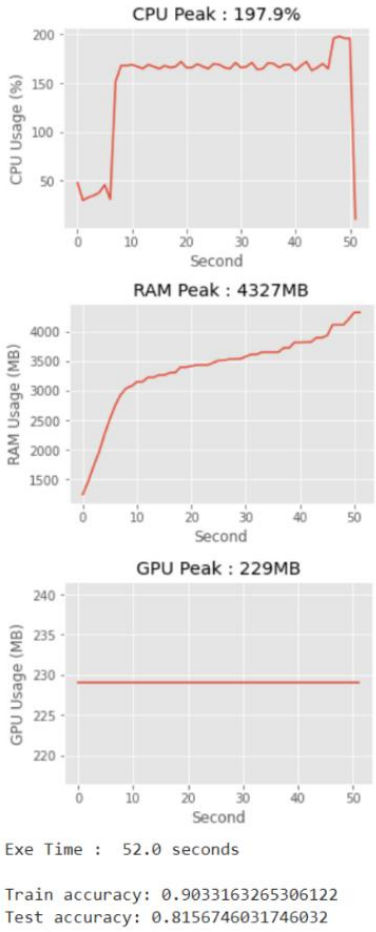

Fig. 19. Results of Initial Architectural Training for a Combination of CNN and ELM.

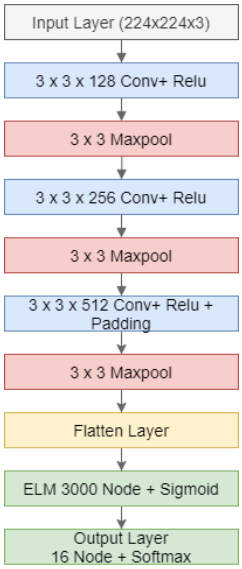

Fig. 20. Combined Architecture of CNN and ELM after Tuning Process.

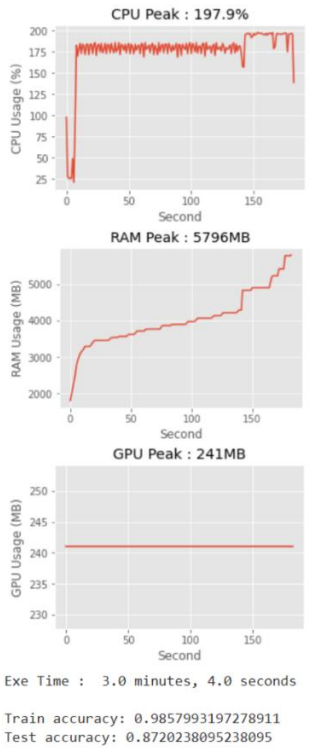

Fig. 21. Results of Tuned Architecture Training from Combination of CNN and ELM.

In the architecture that has been tuned the training process above, the training time is 3 minutes 4 seconds, with peak resource usage of 197.9% CPU, 5796 MB RAM, and 241 MB GPU. In training, we obtain accuracy of 0.985 while the test data was 0.872.

D. Testing and Evaluation Results

The model that has been made in the previous process will be tested with a test scenario that has been made, with several aspects and factors, to find out how well the model is performing.

1) Testing training speed and resource usage: In this test, the model will be tested on how long training time and how large resource use are associated with accuracy, with the following factors:

- The amount of data

This factor is tested to determine how much influence the amount of data has on the training process, by increasing the amount of data from 1,050 per class to 1,400 data per class so that the total data becomes 22,400.

- Variation of the Extraction Layer

In this factor, tests are carried out to determine how much influence the complexity of the extraction layer has on the training process. At this stage an additional layer of convolutional extraction is added to the architecture.

- Number of hidden layers

This factor is tested to determine how much influence the number of hidden layer classification on the training process, on normal CNN plus one hidden layer. In the combination model of CNN and ELM, this stage is not carried out because ELM only has one hidden layer.

- The number of hidden layer nodes

This factor is tested to determine how much influence the number of hidden layer nodes has on the classification process of the training process. In normal CNN the third hidden layer is increased from 512 to 1024 nodes. For the combination of CNN and ELM model hidden nodes increased from 2500 to 300 nodes. After conduting experiment using above factors, the results of this process can be seen in Table II.

2) Cross validation evaluation: The next scenario is evaluation with the cross-validation method. This process is carried out to evaluate the accuracy of the two models that have been made against the training data. This research will use 5-fold cross validation, which means that the training data will be divided into five parts. This evaluation is shown in Table III.

The results above, when plotted with the line chart, are shown in Fig. 22.

TABLE II. RESULTS OF TESTING TRAINING SPEED AND RESOURCE USAGE

Model	Factor	*Training Time*	Resource Usage (Peak)	Accuracy
Normal CNN	Amount of data	6 minutes 3 seconds	CPU 158.9%, RAM 3233MB, GPU 771MB	Train: 0.97 Test: 0.89
	Variation layer extraction	2 minutes 57 seconds	CPU 118.9%, RAM 2662MB, GPU 432MB	Train: 0.96 Test: 0.88
	Number of hidden layers	4 minutes 29 seconds	CPU 153.9%, RAM 3301MB, GPU 771MB	Train: 0.97 Test: 0.91
	Number of hidden layer nodes	6 minutes 2 seconds	CPU 140.9%, RAM 2483MB, GPU 753MB	Train: 0.96 Test: 0.89
Proposed Combination of CNN and ELM	Amount of data	4 minutes 14 seconds	CPU 197.9%, RAM 7074MB, GPU 259MB	Train: 0.97 Test: 0.86
	Variation layer extraction	1 minutes 41 seconds	CPU 197.9%, RAM 5753MB, GPU 259MB	Train: 0.98 Test: 0.85
	Number of hidden layer nodes	3 minutes 49 seconds	CPU 197.9%, RAM 6255MB, GPU 241MB	Train: 0.98 Test: 0.86

TABLE III. RESULTS 5-FOLD CROSS VALIDATION OF NORMAL CNN

Iteration	Accuracy
Iteration 1	0.87
Iteration 2	0.89
Iteration 3	0.90
Iteration 4	0.88
Iteration 5	0.90
Average	0.89

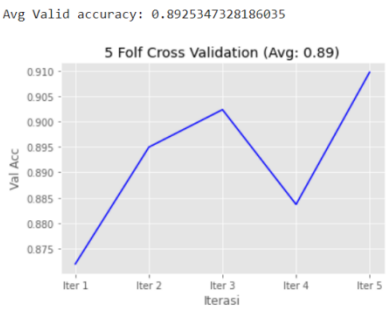

Fig. 22. Plot of Results 5-Fold Cross Validation Normal CNN.

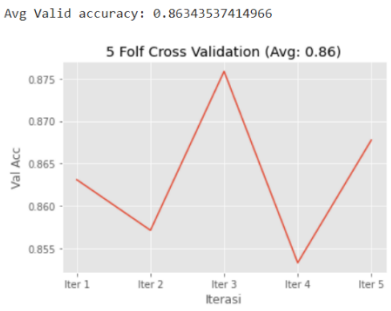

Fig. 23. Plot of Result 5-Fold Cross Validation Combination of CNN and ELM.

The results of the evaluation of the combined CNN and ELM models can be seen in the following Table IV. The results, when plotted with the line chart, are shown in Fig. 23.

3) Accuracy, precision, and recall evaluation: The last scenario is the evaluation of accuracy, precision, and recall of data testing using confusion matrix, this is done to find out how well the model can generalize knowledge.

In the normal CNN model the results of confusion matrix can be seen in the following Fig. 24.

From confusion matrix, accuracy, precision, and recall can be calculated. The results can be seen in the following Table V.

In Table V, the precision value is obtained with a Micro Average of 0.92 and an Average Macro of 0.92. On the other hand, the recall value with a Micro Average of 0.92 and an Average Macro of 0.92.

Average micro calculates the metric independently for each class and then takes the average, suitable for cases with a balanced amount of data for each class. Whereas Average Macro represents the contribution of all classes as whole to calculate the metric mean, it is suitable for cases with a balanced amount of data.

For the combination of CNN and ELM model, the results of the confusion matrix can be seen in the Fig. 25.

From confusion matrix, accuracy, precision, and recall can be calculated, the results of which can be seen in the following Table VI.

In the Table VI, the precision value obtained with Avg Micro is 0.88 and Avg Macro is 0.88. On the other hand, the recall value with Avg Micro was 0.88 and Avg Macro was 0.88.

TABLE IV. RESULTS OF 5-FOLD CROSS VALIDATION COMBINATION OF CNN AND ELM

Iteration	Accuracy
Iteration 1	0.86
Iteration 2	0.85
Iteration 3	0.87
Iteration 4	0.85
Iteration 5	0.86
Average	0.86

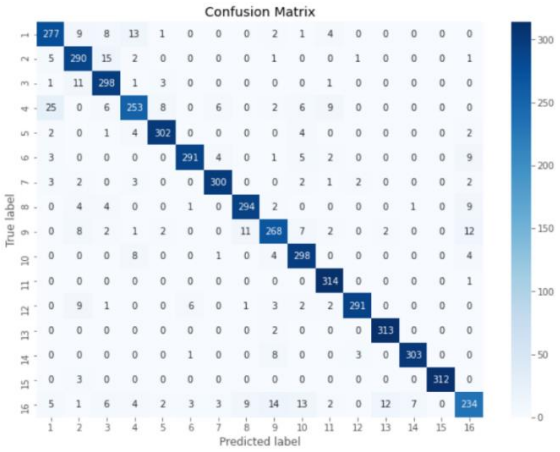

Fig. 24. Confusion Matrix Normal CNN Model.

TABLE V. TABLE NORMAL CNN ACCURACY, PRECISION, AND RECALL RESULTS

Accuracy	0.92	
Class	Precision	Recall
Military Helicopter	0.86	0.88
Armored Car	0.86	0.92
Military Tank	0.87	0.95
Military Jet	0.88	0.80
Military Ship	0.95	0.96
Pistol	0.96	0.92
Military Rifle	0.96	0.95
Grenade	0.93	0.93
Military Box	0.87	0.85
Military Knife	0.88	0.95
Military Helmet	0.93	1.00
Military Binoculars	0.98	0.92
Military Boot	0.96	0.99
Military Bag	0.97	0.96
Army	1.00	0.99
Non-Military	0.85	0.74
Avg Micro	0.92	0.92
Avg Macro	0.92	0.92

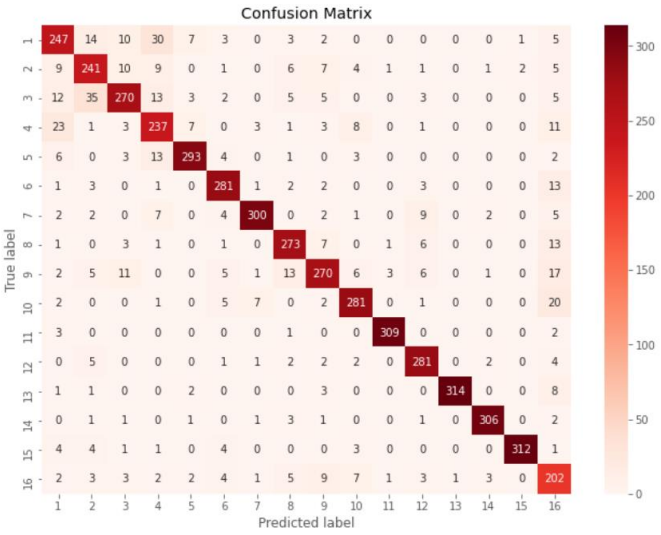

Fig. 25. Confusion Matrix Combination of CNN and ELM Model.

TABLE VI. TABLE COMBINATION OF CNN AND ELM ACCURACY, PRECISION, AND RECALL RESULTS

Accuracy	**0.87**	
Kelas	**Precision**	**Recall**
Military Helicopter	0.78	0.77
Armored Car	0.77	0.81
Military Tank	0.86	0.76
Military Jet	0.75	0.80
Military Ship	0.93	0.90
Pistol	0.89	0.92
Military Rifle	0.95	0.90
Grenade	0.87	0.89
Military Box	0.86	0.79
Military Knife	0.89	0.88
Military Helmet	0.98	0.98
Military Binoculars	0.89	0.94
Military Boot	1.00	0.95
Military Bag	0.97	0.97
Army	0.99	0.95
Non-Military	0.64	0.81
Avg Micro	**0.88**	**0.88**
Avg Macro	**0.88**	**0.88**

E. Analysis and Discussion

Based on the training results in Table II, in the factor of extraction layer variation, one additional convolutional extraction layer and one max pooling layer are added to the architecture. This factor evaluates how much influence the complexity of the extraction layer has on the training process. It is found that the combination model of CNN and ELM achieves processing time 1 minute 43 seconds, which is faster than the normal CNN model. This is because the addition of the extraction layer affects the number of kernels that must be trained iteratively. The effect is that the learning time in the normal CNN model is getting longer, whereas in the combination model CNN and ELM does not carry out a repetitive weight updating process. Therefore, the number of extraction layers does not really affect the combination model of CNN and ELM. For resource usage, the combination CNN and ELM models use 79% more resources on CPU than normal CNN models. The combined CNN and ELM models use 3091 MB more resources on RAM than normal CNN models. The normal CNN model use 176 MB more resources on GPUs compared to the combined CNN and ELM models, gradually. In the training data, the combined CNN and ELM model has a higher accuracy than 0.01 normal CNN model, while the CNN normal model test data is 0.03 superior to the CNN and ELM combination model.

In the factor of the number of hidden layers, it evaluates how much influence the number of hidden layer classifications has on the training process. In this factor, it is only tested on the normal CNN model because the combination model of CNN and ELM only has one hidden layer. It is found that the leaning time in the Normal CNN model is 2 minutes 20 seconds longer than before the addition of the hidden layer, as well as the previous factor, such as the addition of the number of hidden layers has an effect on the amount of weight that must be trained iteratively. The effect is that the tilt velocity in the normal CNN model is getting slower. For resource usage, CPU has 30.1% more resources than without adding hidden layers, normal CNN model RAM uses 1 MB more resources than without adding hidden layers, on normal CNN GPUs the number of resources is the same as before adding hidden layers. In training and testing data, the normal CNN model has smaller accuracy of 0.01 compared to with CNN without the addition of a hidden layer.

In the number of hidden layer nodes factor, it evaluates how much influence the number of hidden layer nodes has on the classification process of the training process. In the third normal CNN, hidden layer is increasing from 512 to 1024 nodes. On the other hand, in the combination model CNN and ELM, hidden nodes are increased from 3000 to 3500 nodes. It is found that the combined model of CNN and ELM require processing time as long as 2 minutes 13 seconds faster than the normal CNN model. This is because the increase in the number of nodes affects the number of weights that must be trained iteratively. Consequently, the leaning time in the normal CNN model is getting longer, while in the combination of CNN and ELM does not perform a repeated weight updating process. Therefore, the number of extraction layers does not really affect the combination model of CNN and ELM. For resource usage, the combination of CNN and ELM models uses 57% more resources on CPU than normal CNN models, combined CNN and ELM models use 3772 MB more resources on RAM than normal CNN models, normal CNN model uses 512 MB more resources on GPUs compared to combined CNN and ELM models. In the training data, the combination of CNN and ELM models have an accuracy of 0.02 which is superior to the normal CNN normal, while the normal CNN model achieve accuracy in test data around 0.03 which is superior to the combination of CNN and ELM models.

The results of the cross-validation evaluation in Tables III and IV show that the average validation accuracy of the normal CNN model is superior, namely 0.89 compared to the average validation accuracy in the combined CNN and ELM model, which is 0.86. It can be seen that both models produce fairly even accuracy. In each part of the cross-validation evaluation process.

For the evaluation of accuracy, precision, and recall, the results are obtained in Tables V and VI. Both from the accuracy, precision and recall of normal CNN models are superior to the combination of CNN and ELM models. This indicates that the normal CNN model has a better generation capability, but with a single layer and without the weight updating process the combination of CNN and ELM has produced very good performance as well. If we look further at the results' confusion matrix on the combination of CNN and ELM model, the prediction error occurs in objects that have many features, helicopters with aircraft and armored cars with tanks. It can be seen that the multilayer FCL on CNN has better ability in the pattern features that are similar or complex compared to a single layer in ELM.

V. Conclusions

From the research process that has been implemented, several conclusions can be drawn as follows:

- The combined CNN and ELM model uses a convolutional extraction layer on CNN, which is then combined with the classification layer using the ELM method. The model learning time is always shorter, approximately 2 minutes, compared to normal CNN. It is because the normal CNN uses full connected layer (FCL) based backpropagation, which still uses slow gradient-based learning algorithms to carry out learning.
- The normal CNN model resource usage is 57% smaller on CPU resources and uses an average of 3568 MB of smaller resources on RAM, but the combined CNN and ELM models uses 400 MB of smaller resources on GPUs.
- Accuracy, precision and recall of normal CNN models are slightly higher by 0.03 to 0.04 compared to combined CNN and ELM models. However, with one layer and without updating process, the combined weight of CNN and ELM was maintaining the accuracy.

References

[1] S. Liu and Z. Liu, “Multi-Channel CNN-based Object Detection for Enhanced Situation Awareness,” pp. 1–9, 2017, [Online]. Available: http://arxiv.org/abs/1712.00075.

[2] E. Prasetiawan, “Implementation of Distinction Principles Related to Civil and Military Object in Indonesia (in Bahasa Indonesia),” Universitas Airlangga, 2019.

[3] M. Sharma, A. Bhave, and R. R. Janghel, “White Blood Cell Classification Using Convolutional Neural Network,” in Soft Computing and Signal Processing, 2019, pp. 135–143.

[4] Y. A. Hambali, “C # Based Process Area Application using Visual Studio (in Bahasa Indonesia),” Ilmu Komput., p. 14, 2011.

[5] G. Bin Huang, Q. Y. Zhu, and C. K. Siew, “Extreme learning machine: A new learning scheme of feedforward neural networks,” IEEE Int. Conf. Neural Networks - Conf. Proc., vol. 2, pp. 985–990, 2004, doi: 10.1109/IJCNN.2004.1380068.

[6] Z. Yang et al., “Deep transfer learning for military object recognition under small training set condition,” Neural Comput. Appl., vol. 31, no. 10, pp. 6469–6478, 2019, doi: 10.1007/s00521-018-3468-3.

[7] T. Hiippala, “Recognizing military vehicles in social media images using deep learning,” 2017 IEEE Int. Conf. Intell. Secur. Informatics Secur. Big Data, ISI 2017, pp. 60–65, 2017, doi: 10.1109/ISI.2017.8004875.

[8] F. Mahmud and M. Al Mamun, “Facial Expression Recognition System Using Extreme Learning Machine,” Int. J. Sci. Eng. Res., vol. 8, no. 3, pp. 266–267, 2017, [Online]. Available: http://www.ijser.org.

[9] A. R. Wiyono, “Introduction to Face Expression Image Using Principal Component Analysis (PCA) and Extreme Learning Machine Algorithm (in Bahasa Indonesia),” Jurnal Ilmiah Matermatika (MATH), vol. 6, no. 2, pp. 2–6, 2018.

[10] Y. Bazi and F. Melgani, “Convolutional SVM Networks for Object Detection in UAV Imagery,” IEEE Trans. Geosci. Remote Sens., vol. 56, no. 6, pp. 3107–3118, 2018, doi: 10.1109/TGRS.2018.2790926.

[11] F. Hu, G. S. Xia, J. Hu, and L. Zhang, “Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery,” Remote Sens., vol. 7, no. 11, pp. 14680–14707, 2015, doi: 10.3390/rs71114680.

[12] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, “Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification,” Ieee Tgrs, vol. 55, no. 2, pp. 645–657, 2016, doi: 10.1109/TGRS.2016.2612821.

[13] N. Sharma, V. Jain, and A. Mishra, “An Analysis of Convolutional Neural Networks for Image Classification,” Procedia Comput. Sci., vol. 132, no. Iccids, pp. 377–384, 2018, doi: 10.1016/j.procs.2018.05.198.

[14] L. Deng and D. Yu, “Deep Learning: Methods and Applications,” Found. Trends®in Signal Process., vol. 7, no. 3–4, pp. 197–387, 2014, doi: 10.1561/2000000039.