EXTENDED EXTREME LEARNING MACHINE: FRAMEWORK AND APPLICATIONS

BOONNITHI JIARAMANEEPINIT

A THESIS REPORT SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF ENGINEERING IN COMPUTING IN ENGINEERING SYSTEM
INTERNATIONAL COLLEGE
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
ACADEMIC YEAR 2017
KMITL-2017-IC-M-11-04

Extended Extreme Learning Machine: Framework and Applications

Boonnithi Jiaramaneepinit

Master of Engineering in Computing in Engineering System
International College
King Mongkut's Institute of Technology Ladkrabang
Academic Year 2017
KMITL-2017-IC-M-11-04

Abstract

Machine learning has been one of the focus topics in research field for several real-world applications. One of the well-known machine learning systems is neural network. It becomes popular in recent decades because its complexity can achieve satisfied results in several tasks. Generally, every machine learning approach has its learning algorithms. For neural network, conventional method in training is backward propagation of errors, or backpropagation in short. Even it is conventional method, it has many drawbacks. Due to its iterative approach, it consumes a lot of computational power and processing time.

In the last decade, Extreme learning machine (ELM) was proposed. It is used for training a feedforward neural network with one hidden layer, also known as single-hidden layer feedforward neural network (SLFN). It minimizes error by computing the weights of a network using a whole dataset in one-shot calculation. This makes it trains the networks in very fast manners. However, it has many difficulties when encounters with high-dimensional data. Because SLFN has no layer in processing data from input layer into higher level features, this makes representation of data does not being well appointed. Furthermore, training using ELM can become instability due to its randomness in training procedure. In order to overcome these difficulties, Extended Extreme Learning Machine (X-ELM) is proposed.

X-ELM extends ELM to predict the final outputs from output of several trained networks. It is based on ensemble approach. In this research, X-ELM is applied to many real-world applications, including classifying vehicle types, vehicle colors, and handwritten digits.

Acknowledgments

I would like to express my deep gratitude to Asst.Prof.Dr. Chaiwat Nuthong, my advisor, for providing me vast knowledge, several resources, and useful assessments. I would like to thank Dr. Ukrit Watchareeruetai also for giving us criticism and advices during the research. Special thanks to Hinagiku Katsura for the encouragement and several forms of support. Thanks to professors, seniors, and juniors at Applied Machine Learning Laboratory and Computer Vision Laboratory at the International College, King Mongkut's Institute of Technology Ladkrabang for all helpful advices and discussions. Furthermore, my grateful thanks are also extended to family and friends for assisting me to make this thesis completed. Lastly, we would like to express sense of gratitude to all, who directly or indirectly, have a part in this venture.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

In the last decades, artificial neural network (ANN) has been developed and used in several tasks, such as regression and classification [1, 2]. Due to its satisfied performance, it becomes one of the most popular systems in research field. Although, the first developed structure of ANN was very simple, rapid grown of its researches lead to more complex structures, i.g. deep learning structures [3, 4].

Even with the complex structure of ANN, the learning algorithm being used is still conventional. Such method is called backward propagation of errors, backpropagation (BP). It adjusts weights between layers in ANN into proper values. Based on gradient descent, it iteratively computes for errors and backpropagate through the network to fix the weights [5, 6]. Although, it is a conventional method, this method still has many drawbacks. One of the well known drawbacks is its slow learning speed. The speed of using BP is found to be slow due to its iterative calculation. As the results, BP consumes a lot of computing resources and takes large amount of time to be completed [7, 8].

In order to conquer such drawbacks, Extreme Leaning Machine (ELM) was proposed by Huang et al. [9]. It approximates weights and maps features to targets based on least-squares method. Moreover, it can train networks very fast and has promising results. However, its performance still has a room for improvement.

1.2 Problem Descriptions

There are many advantages of using ELM instead of BP. First is that it has a better generalization performance due to small norm of weights [10]. Furthermore, it minimizes errors based on least-squares solution. However, for some tasks given high dimensional data, such as image classification, ELM still shows a room for improvement [11]. As ELM is originally used for training single-hidden layer feedforward neural network (SLFN), the structure has limitations which extracting data into high level features. This leads to an incomplete representation of data in the network. Nevertheless, the performances using ELM can be frustrated because of arbitrary initial weights.

1.3 Research Objectives

This research studies frameworks of ANN and conducts experiment on both ELM and the proposed framework, namely Extended Extreme Learning Machine (X-ELM). Then the results are compared. The objectives to be achieved are set as follows:

- To compare performance of ELM trained on SLFN and the new framework
- To compare performance trained on vehicle type, vehicle color, and hand written digits datasets
- To study the effectiveness of integrating ELM into real world applications
- To study the effectiveness of applying ELM for more complex ANN structure

1.4 Scope of the Study

This research focuses on constructing and measuring performance of a new neural network framework. The research is conducted on three datasets, i.e. vehicle color, vehicle type, and hand written digit classifying tasks. Performance measurement of each task is solely performed multiply times, so that statistic values of performance can be calculated. In addition, performance is measured in term of both accuracy and computational time. The resources for vehicle color and type dataset in this research are captured and processed by Saripan et al. [**?**]. On the other hand, the hand written digits dataset is the well known Modified National Institute of Standards and Technology (MNIST) database.

1.5 Expected Contribution

This thesis proposes a novel neural network framework, Extended Extreme Learning Machine (X-ELM), which shall be used for achieving better performance in real world application. It shall circumvent the previously mentioned difficulties of ELM.

Chapter 2

Literature Review

This chapter is devoted for literature review, which includes past studies, related works and theories that involve in this research. There are two main topics to be discussed: machine learning and real world application. Detailed information can be found in following sections.

2.1 Machine Learning

Machine learning is an ability of a computing software to solve a problem or do a task without explicitly programmed to do so. Moreover, it is also considered as in a boundary of artificial intelligence researching field. There are several computing system available for machine learning. However, in this research, ANN is solely focused. This includes single-hidden layer feedforward neural network model and multi-hidden layer feedforward neural network. The details of these models and their learning algorithm will be described in following subsections.

2.1.1 Single-Hidden Layer Feedforward Neural Network

A single-hidden layer feedforward neural network (SLFN) is one of the simplest architectures of ANN. It is directed, and there are no cycles in it. For this architecture, data as signal moves in one direction, from input layer to hidden layer, and from hidden layer to output layer, as shown in Figure 2.1.

Each node in hidden layer sum all products of nodes' output from the previous layer with cor-

Figure 2.1: A single hidden layer feedforward neural network

responding weights. Furthermore, the summed signal is sent to an activation function in order to create non-linearity inside the network. This allows the predictive models be able to learn more complex data. The other main components of ANN are activation function and learning algorithm. The traditional algorithm is backpropagation. It has a huge drawback which is its iteratively computing based on gradient descent [12].

2.1.2 Multi-Hidden Layer Feedforward Neural Network

A multi-hidden layer feedforward neural network is another architectures of ANN. Generally, it has more complexity than SLFN. As is is feedforward, it is also directed, and there are no cycles inside. For this architecture, similar to SLFN, data moves in one direction, from input layer to first hidden layer, and from first hidden layer to second hidden layer, and so on until it reach output layer.

2.1.3 Extreme Learning Machine

Extreme Learning Machine (ELM) was proposed as learning algorithm in the last decade. It has a generalization performance for SLFN based on least-square error. According to Huang et al., this learning algorithm is able to solve three issues, which are intensive human intervention, slow learning speed, and poor learning scalability [13]. In addition, by training SFLN using ELM also helps in decrease training time significantly.

There are several classification tasks that work well when using ELM, i.e., optical character classification (OCR) and traffic sign classification [14, 15]. Although ELM shows results that outperform other approach in many applications, there is still room for the improvement. Thus, this study focuses on constructing and measuring performance of neural network's framework trained with based on ELM. Details are discussed in the following chapter.

2.2 Real World Application

This work focuses on constructing architecture for real world tasks' classifiers, including classifiers in traffic surveillance and OCR system. The classification in traffic surveillance system divides into two main tasks. One of the tasks is vehicle type classification, and another one is vehicle color classification. The classification experiment for OCR system is hand written digits.

2.2.1 Vehicle Classification in Traffic Surveillance

Traffic surveillance cameras are set in several metro areas nowadays. They have several objectives. One of theirs objectives is to monitor unexpected circumstances. Other is to search for vehicles in traffic system. There is many cases where staffs or police officers spend time on finding some targeted vehicles by watching streaming video through the monitor. This takes a lot of effort. Moreover, it could lead staffs to eye fatigue. Nevertheless, with inappropriate environment low searching accuracy might be gotten [16]. For these reasons, an automatic vehicle classifying system are studied.

In the past, there are many proposed systems. Chang et al. applied deep convolutional neural network for capturing high level features then doing classification of vehicles [17]. Saripan et al. used decision tree based approaches to classify vehicle's characteristic [18]. However, most of them needs huge amount of resources or consumes a lot of training time. For this reason, this study proposes and applies frameworks for classifications which have high accuracy and consume small amount of training time.

2.2.2 Modified National Institute of Standards and Technology

The Modified National Institute of Standards and Technology (MNIST) database is a database of handwritten digits. It is a sub-dataset of National Institute of Standards and Technology (NIST) database. The image of digits has been centered and normalized into same size. There are 60,000 training samples and 10,000 testing samples. As the task of MNIST is to recognize digit characters, MNIST is widely used for training, validating, and testing models in machine learning research field. Details of the dataset will be explained in Section 5.4.

Chapter 3

Background Knowledge

This chapter introduces various notions and concepts that is used in this research. Concise explanation, including fundamental information, will pointed out.

3.1 Single-Hidden Layer Feedforward Neural Networks

A single-hidden layer feedforward neural network (SLFN) is a architecture of neural network which contains a hidden layer of hidden nodes. It composes of processing units and directed weight, which are same as typical ANN.

Processing units or simple processing units can also be called nodes. Output of each node pass to an activation function. The resulted signal is then send to next corresponding nodes layer in the network. Equation (3.1) is an well-known activation function, namely sigmoid function. This study is based on this activation function.

$$f(x) = \frac{1}{1 + e^{-t}} \tag{3.1}$$

Another component in SLFN is a learning algorithm. Before SLFN is be able to be used to predict, it requires network's training. In the training, weighted links are adjusted to appropriate values.

For N instances $(\mathbf{x}_i, \mathbf{t}_i)$ where input vector $\mathbf{x}_i = [x_{i1}, x_{i2}, ..., x_{in}]^T \in \mathbb{R}^n$ and expected output vector $\mathbf{t}_i = [t_{i1}, t_{i2}, ..., t_{im}]^T \in \mathbb{R}^m$, SLFNs with $\hat{N}$ hidden nodes are mathematically modeled as

$$\mathbf{y}_j = \sum_{i=1}^{\hat{N}} \boldsymbol{\beta}_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i), \quad \text{for} \quad j = 1, ..., N \tag{3.2}$$

where $\mathbf{w}_i = [w_{i1}, w_{i2}, ..., w_{in}]^T$ is a vector of weighted links from input nodes to i^{th} hidden node, $\boldsymbol{\beta}_i = [\beta_{i1}, \beta_{i2}, ..., \beta_{in}]^T$ is a vector of weighted links from the i^{th} hidden node to output nodes, and b_i is the threshold of the i^{th} hidden node [13].

In training process, $\mathbf{y}_j$ in Equation (3.2) is substituted with expected output vector $\mathbf{t}_j$ and is written as

$$\mathbf{t}_j = \sum_{i=1}^{\hat{N}} \boldsymbol{\beta}_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i), \quad \text{for} \quad j = 1, ..., N \tag{3.3}$$

Equation 3.3 can also be written as

$$\mathbf{T} = \mathbf{H}\boldsymbol{\beta} \tag{3.4}$$

where

$$\mathbf{H} = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \dots & g(\mathbf{w}_{\hat{N}} \cdot \mathbf{x}_1 + b_{\hat{N}}) \\ \vdots & \ddots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_{\hat{N}} + b_1) & \dots & g(\mathbf{w}_{\hat{N}} \cdot \mathbf{x}_{\hat{N}} + b_{\hat{N}}) \end{bmatrix}, \boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_1^T \\ \vdots \\ \boldsymbol{\beta}_{\hat{N}}^T \end{bmatrix}, \mathbf{T} = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{bmatrix}.$$

H is called the hidden layer output matrix. The i^{th} column of **H** is the output of i^{th} hidden node with $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N$ as inputs [9].

In this study, SLFN is used as a simple neural network model in experiment for performance of ELM. Moreover, more complex architecture will be introduced in Chapter 4. The results are compared in experiments.

3.2 Learning Algorithm

A learning algorithm is an algorithm which neural network uses for learning data. The goal of learning is to find proper weights that minimize the error; in other words, maximize the accuracy. There are various learning algorithm that can be applied, for example, gradient descent, algebraic approach, newton's method, etc. However, in this project, only one approach is mainly focus, which is ELM based on algebraic computation.

3.2.1 Extreme Learning Machine

Extreme learning machine (ELM) was originally proposed as a learning algorithm for SLFN [9]. It was claimed that it solves problems of slow speed of gradient learning and the tuning of iterative parameters [19].

In ELM, weights between input layer and hidden layer are randomly generated. Then, weights between hidden layer and output layer are calculated. It means that not all weights need to be calculated. As it is one-shot calculation, it does not need an iterative computation, like BP.

Generally, ELM is processed by four following steps:

1. Arbitrary generate weight $\mathbf{w}_i$ between input layer and hidden layer
2. Arbitrary generate bias b_i
3. Compute for the output matrix $\mathbf{H}$ of hidden layer
4. Compute the output weights β using Equation (3.5)

$$\beta = \mathbf{H}^{\dagger}\mathbf{T} \tag{3.5}$$

For the SLFN, the objective of a learning algorithm is that it need to find $\mathbf{w}_i$, $\mathbf{b}_i$, and β such that $||\mathbf{H}\beta - \mathbf{T}||$ becomes minimum as much as possible; in other words, to minimize an error. The only weight that needs an adjustment is weight β. Then the solution based on smallest norm of least-squares error is calculated from Equation (3.5) where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of $\mathbf{H}$. Comparing ELM to BP, ELM has several advantages. Following is a list of main advantages:

1. No iterative calculation
2. A unique solution based on minimum norm least-squares error
3. Minimum training error regardless of local minimum
4. Generalization performance due to smaller norm of weights [20]

Raw Feature	**One Hot Encoded Features**		
	Feature 1 (“Apple”)	Feature 2 (“Banana”)	Feature 3 (“Orange”)
“Apple”	1	0	0
“Banana”	0	1	0
“Orange”	0	0	1

Table 3.1: Examples of transforming a raw feature into one hot encoded features

3.3 One Hot Encoding

One hot encoding is a way to transform a categorical discrete feature into zero-one scheme features. In sklearn documentation [21], it said that “Encode categorical integer features using a one-hot aka one-of-K scheme”. For example, if the feature categorizes into “Apple”, “Banana”, and “Orange”, “Apple” would be encoded into (1, 0, 0), while “Banana” would be encoded into (0, 1, 0) and “Orange” would be encoded into (0, 0, 1). This is also shown in Table 3.1.

Chapter 4

Methodology

This chapter introduces the methods use and conduct in this study. The first section, Extended Extreme Learning Machine, will explain about the architectures models for the experiments. The following three sections, i.e. vehicle type, color classification, and MNIST using neural networks, will describe parameters tuned for each dataset.

4.1 Extended Extreme Learning Machine

Extended Extreme Learning Machine (X-ELM) uses ELM as an extension in order to predict the final outcome. Based on ELM, X-ELM attaches one or more predictive models together by extending neural network trained with ELM. The architecture of a neural network based on X-ELM can be depicted as shown in Fig. 4.1. In this research, predictors are referred to regressors and classifiers of an any model.

As depicted in Fig. 4.1, X-ELM composed of three main components, i.e. input layer, hidden layer, and output layer, which are seem to be same as conventional SLFN. However, it differs inside the layer. X-ELM takes predictive models to the input layer. Models $m_1, m_2, \ldots, m_M$ in Fig. 4.1 illustrate those aforementioned predictive models. Another two layers are same as SLFN.

X-ELM framework can be divided into two cases. First is a case which all models are neural networks. Figure 4.2 illustrates such a case. It contains P neural networks with M hidden nodes and R hidden nodes for the extended network. This can be denoted as (P $\times$ M, R). Second is a case which predictive models are differed. Figure 4.3 depicts such a case.

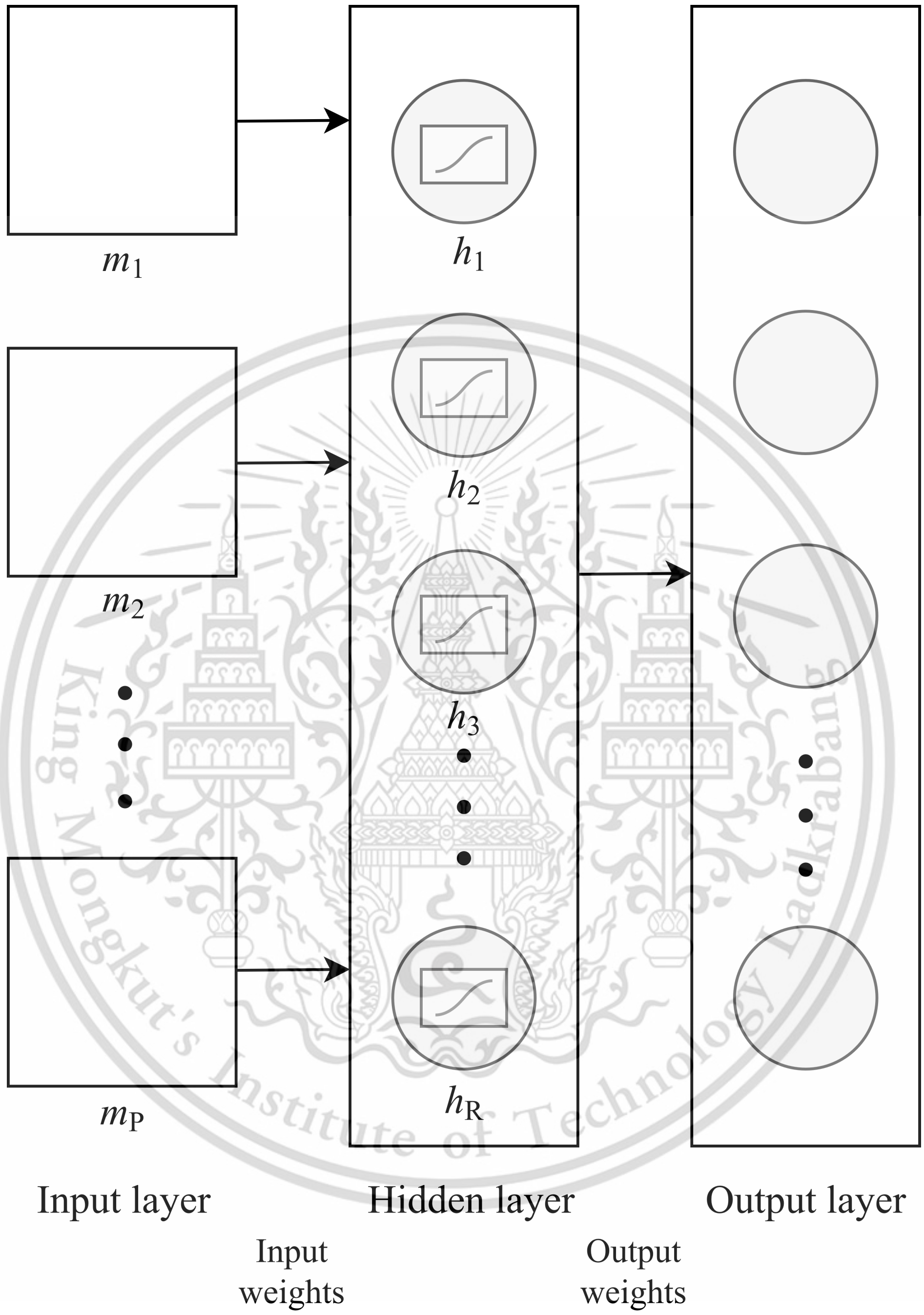

Figure 4.1: Extended Extreme Learning Machine

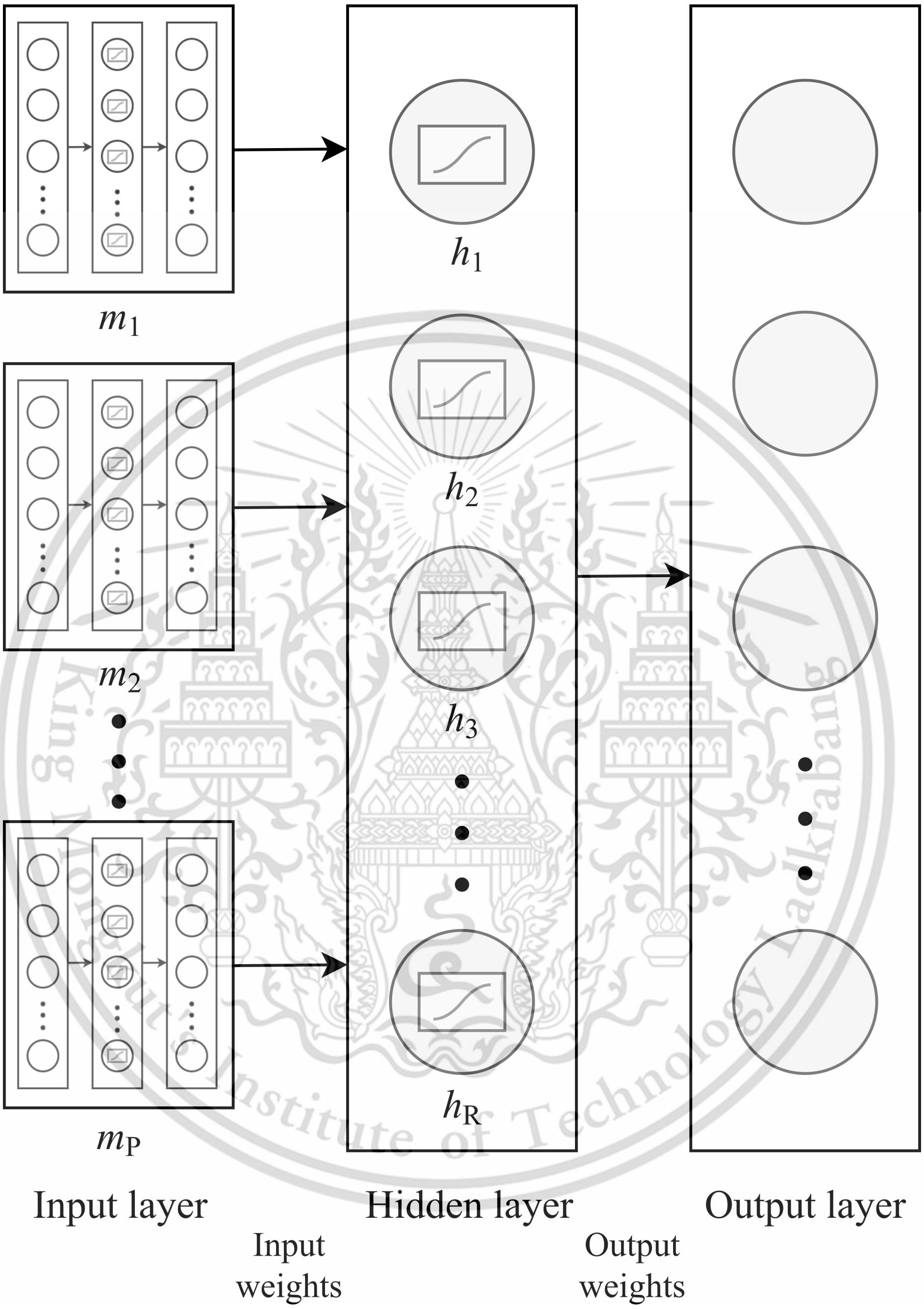

Figure 4.2: Extended Extreme Learning Machine framework with neural networks as all predictors

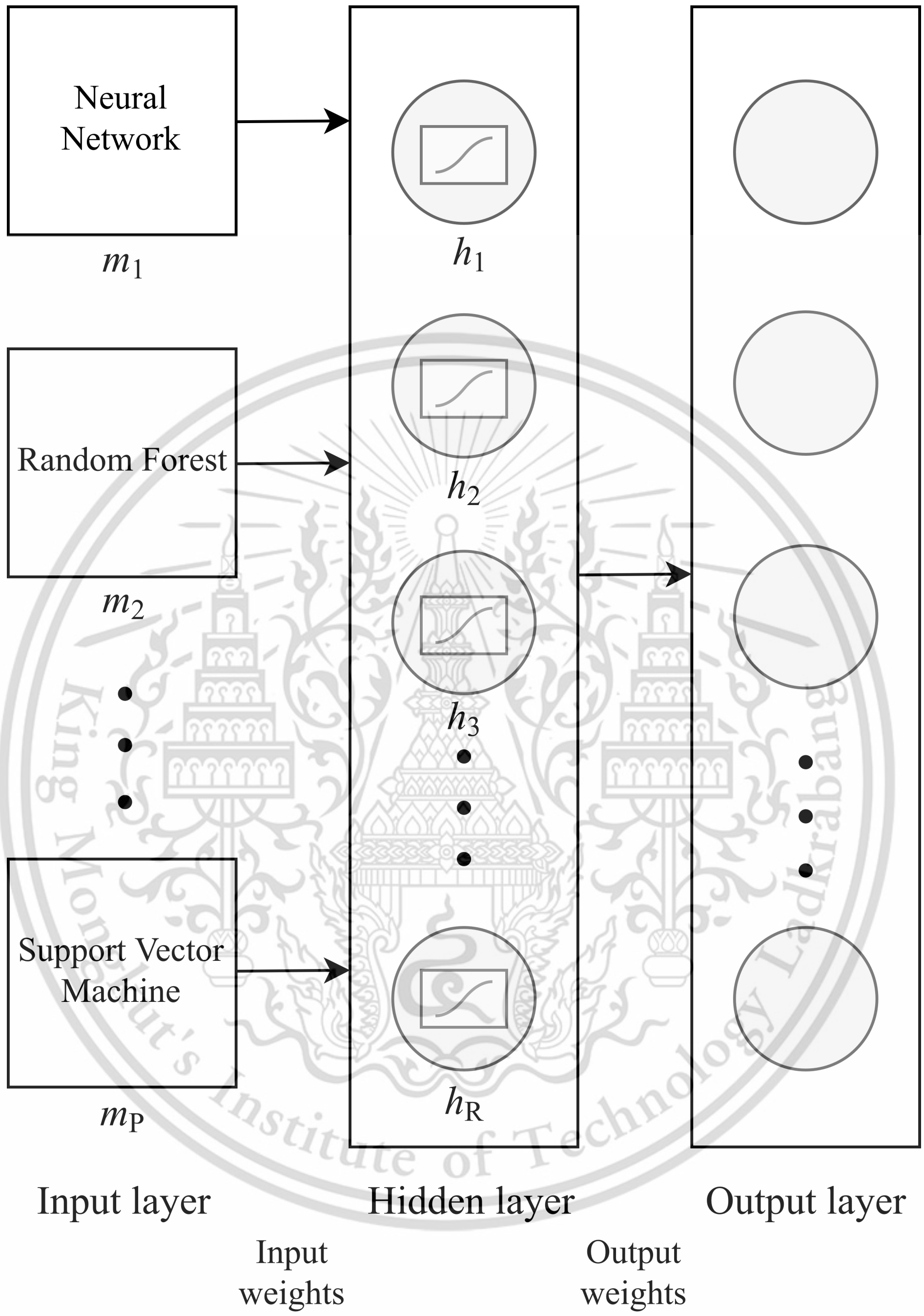

Figure 4.3: Extended Extreme Learning Machine framework with multiple machine learning approaches as predictors

4.1.1 Procedures of Constructing X-ELM

The steps of modeling and training neural network based on X-ELM are shown as in the following:

1. Model predictors (can be any number of regressors and classifiers, i.e. artificial neural network, random forest, support vector machine, etc.)
2. Train the modeled predictors with a whole training dataset or subset of training dataset
3. Feed a training dataset or subset of training dataset to all trained predictors
4. Combine their outputs by putting them together into a matrix (one-hot encoding can be applied if necessary)
5. Use this matrix as an input for training an extended network using ELM
6. Follow steps of ELM in training the extended part of neural network
7. Make final decision based on classification probabilities of each class

Note that each predictor can be tuned as weak learners as well.

4.2 Classifying Tasks

In this study, there are 3 tasks designed for measuring performance of different approaches. The 3 tasks are vehicle type classifying task, vehicle color classifying task, and MNIST OCR task. The overview of each task will be described in following subsections.

4.2.1 Vehicle Type Classification

Vehicle type classification for traffic surveillance system is a process of classifying input data into a type of vehicle. Instead of using image as an input, in the study, preprocessed dataset is used. There are several techniques that can be used for solving a classification problem. However, neural network is the only focused system in this study. The types are encoded using one hot encoding before being trained in neural network.

4.2.2 Vehicle Color Classification

Vehicle color classification is a process of classifying input data into a color. Similar to vehicle type classification, the dataset used for conducting experiments for vehicle color classification is also preprocessed as well.

4.2.3 Modified National Institute of Standards and Technology Optical Character Recognition

Modified National Institute of Standards and Technology (MNIST) Optical Character Recognition (OCR) is a process of classifying number digits, ”0” to ”9”. The input to predictors is an image with size of 28 by 28 pixel. The output is the label of number digits. To make the dataset works in classification using neural network, one hot encoding is applied for the output.

Chapter 5

Experimentation and Results

First, in this section, there will be explanations and lists of environment and development tools used in this research. Furthermore, processes of development each classifier are further pointed out individually according to the procedures of each computing system. Last, the results achieved in the experiment are summarized and discussed.

5.1 Development Environment

Experiments in this work are all conducted in MATLAB R2017b. Moreover, the experiments are performed on a computer with specifications mentioned in Table 5.1. Additional MATLAB's functions used in the experiments are shown in Table 5.2. Note that MATLAB allows maximum size array of 30256 megabytes.

Processor	AMD Ryzen 7 1700X Eight-Core Processor, 3500 MHz, 8 Cores, 16 Logical Processors
Physical Memory Capacity	16.0 GB
Physical Memory Type	DDR4
Physical Memory Speed Bus	2133 MHz
Operating System	Microsoft Windows 10 Pro 10.0.16299 Build 16299
Solid State Drive	SanDisk Ultra II 480GB

Table 5.1: Specifications of the experimental computer

External MATLAB's Function Name	Purpose
ELM	Train and test based on ELM approach
loadMNISTImages	Load and return a 28 by 28 by [number of images] matrix containing the raw MNIST images
loadMNISTLabels	Load and return a vector with a size of [number of images] containing the labels for the MNIST images

Table 5.2: List of additional third party MATLAB's function

5.2 Experiment 1: Vehicle Type Classification

5.2.1 Objective

This experiment is conducted to measure and compare accuracies and time consumptions during the training process between different learning algorithms for different neural network architectures for classifying a vehicle type. The two approaches experimented are ELM and X-ELM.

5.2.2 Experiment Setup

The experiment is set up as follows:

Dataset: The experiments are conducted with dataset from [18]. There are 914 instances. Each instance consists of 5 attributes and 4 labeled classes. The 5 attributes are X coordinate from top-left corner of an original image, Y coordinate from top-left corner of an original image, height of a processed image, width of a processed image, and area's ratio of vehicle over background of a processed image. The 4 classes are small, medium, large, and unknown. Moreover, the lists of attributes and labeled class mentioned can also be seen in Table 5.3. The data type of the first four attributes are integer of pixels, while the fifth attribute is a floating number. In addition, two third of the dataset is divided to be used as training data and other one third is used in testing. These sets are randomly chosen from the whole dataset.

Attributes	Labeled Classes
X coordinate (from top-left corner)	Small
Y coordinate (from top-left corner)	Medium
Height	Large
Width	Unknown
Area's ratio of vehicle over background	

Table 5.3: List of attributes and labeled classes in vehicle type classification

5.2.3 Evaluation Models

To make a comparison of performance between ELM and X-ELM, four evaluation models are designed. The first three models are based on ELM, while the forth model is based on X-ELM. Table 5.4 shows details and parameters tuned. In this context, hidden nodes are referred to neural nodes in a hidden layer.

In addition, models ELM1, ELM2, and ELM3 have the same framework of training and activation function, they differ in number of hidden nodes inside the network's architecture. The number of hidden nodes are differed for an observing of their effects had on performance.

The forth model, XELM1, is constructed differently. It differs from the first three ELM based models by both framework and network's architecture. It uses X-ELM as a framework and an architecture for having multiple SLFNs. Mark that the models in input layer of X-ELM based framework are actually the model of ELM. This makes the network contained more complexity. Numbers of hidden nodes are defined by two numbers, i.e. a number of hidden nodes for input predictors, which is 10, and a number of hidden nodes for the extended part of the network, which is 40. Nevertheless, number 3 refers to a number of predictors created for input.

5.2.4 Results

The resulted performances are measured in two terms, i.e. testing accuracy and training time. Testing accuracy is measured in percentage. It is calculated by dividing a number of correctly classified vehicle type by a number of all vehicle times one hundred. Thus, the possible range of testing accuracy is from 0 to 100. Furthermore, training time is measured of how long the training

Model	Framework	Architecture	Activation Function	Number of Hidden Nodes
ELM1	ELM	SLFN	Sigmoid	10
ELM2	ELM	SLFN	Sigmoid	50
ELM3	ELM	SLFN	Sigmoid	100
XELM1	X-ELM	Multiple SLFNs	Sigmoid	$(10 \times 3, 40)$*

* There are 3 sub-classification models contained 10 neural nodes in their hidden layer. The extended network contains 40 neural nodes in its hidden layer.

Table 5.4: Information of models used in experiments and evaluation for Experiment 1 and Experiment 2

took to completed the model. This excludes the time used for preprocessing.

The details of results are shown in Table 5.5. For graphical representation of Table 5.5, they are shown in Figure 5.1 and Figure 5.2. In addition, as the initial input weights are arbitrary generated, a hundred time of evaluations are performed to measure the average performance, peak performance, and the stability of a model. Four statistic values are shown, i.e. minimum, mean, maximum, and standard deviation.

As shown in Table 5.5, model XELM1 achieves the better values in every statistic measurement. It shows that it can outperform the original ELM in both higher average and peak accuracies with lower standard deviation. However, it consumes time more than the ELM based models. This may be due to the constructing time of predictors. Although, the training time using X-ELM is higher than ELM, it still take only at most around 80 milliseconds. Therefore, in this experiment, X-ELM shows a better accuracy than ELM with acceptable amount of time using for training.

5.2.5 Summary

In This experiment, classification dataset of vehicle type is used. ELM achieves an average accuracy of 83.35% with standard deviation of 1.78. The maximum accuracy of 87.21% is achieved by ELM. For X-ELM, it outperforms accuracy of ELM in all statistic measurements. The higher average accuracy of 83.85% with lower standard deviation of 1.69 is achieved. Moreover, the peak accuracy of X-ELM experimented is 89.18%. On the whole, X-ELM perform better accuracies and

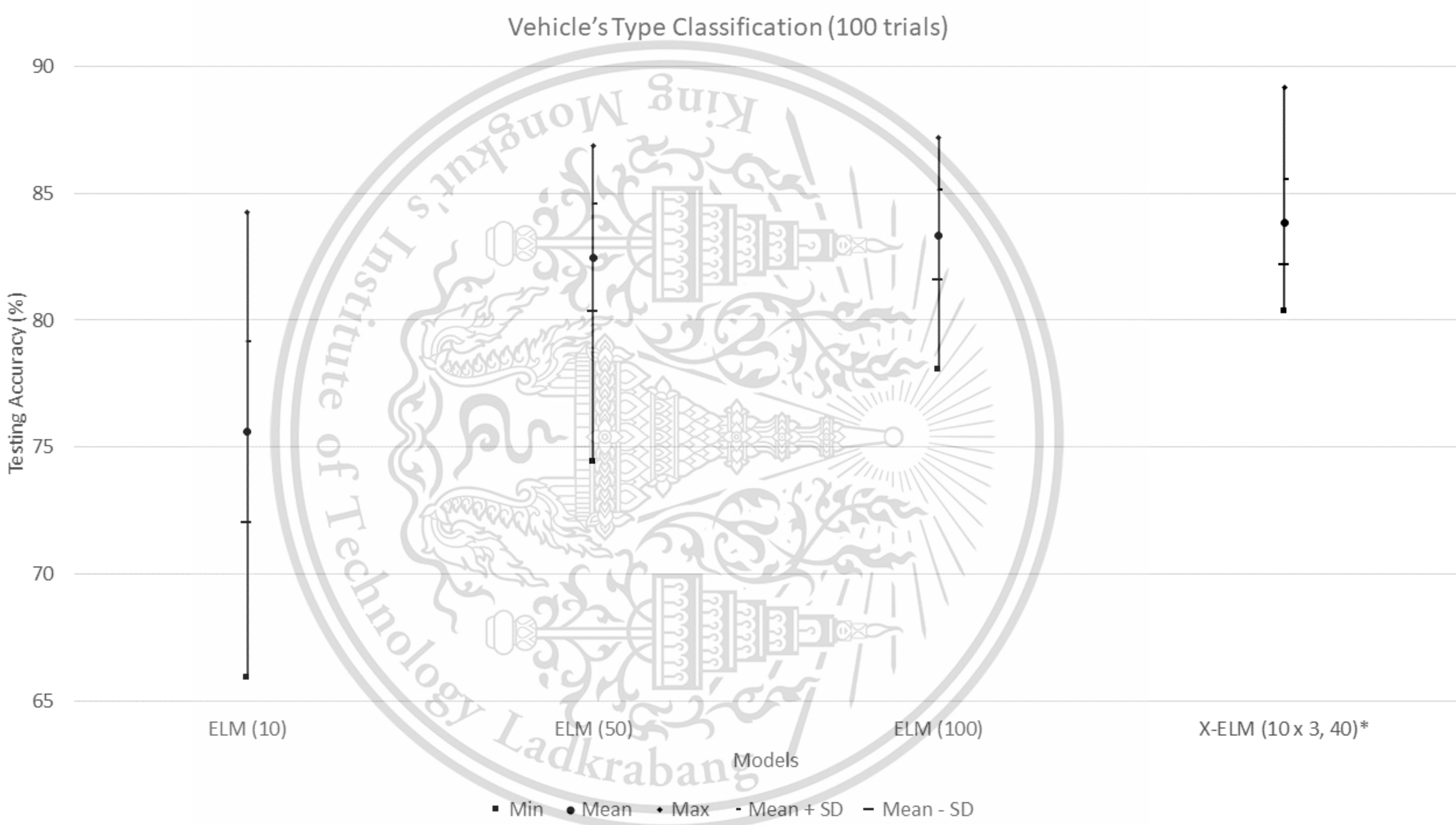

Figure 5.1: Comparison between models of all statistical measurements of testing accuracy on vehicle's type classification

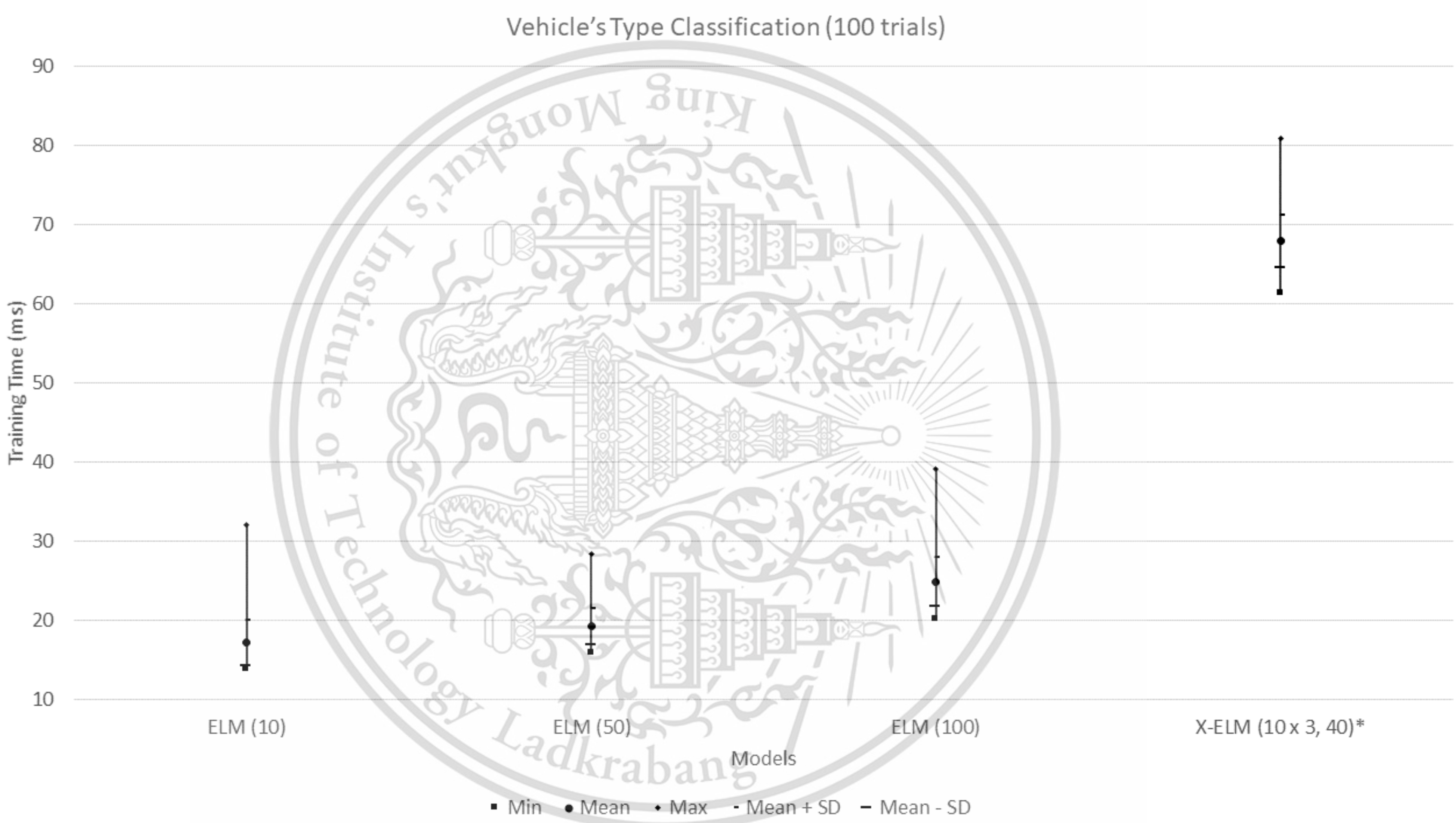

Figure 5.2: Comparison between models of all statistical measurements of training time on vehicle's type classification

Model	Testing accuracy (%)				Training time (ms)			
	Min	Mean	Max	SD	Min	Mean	Max	SD
ELM1	65.90	75.59	84.26	3.55	**13.78**	**17.13**	32.01	2.84
ELM2	74.43	82.46	86.89	2.12	15.93	19.22	**28.39**	**2.30**
ELM3	78.03	83.35	87.21	1.78	20.07	24.81	39.07	3.06
XELM1	**80.33**	**83.85**	**89.18**	**1.69**	61.26	67.85	80.92	3.36

Table 5.5: Comparison table of testing accuracy and training time of three ELM based models and one X-ELM based model for vehicle type classifying task

lower standard deviation of each run.

5.3 Experiment 2: Vehicle Color Classification

5.3.1 Objective

This experiment is conducted to measure and compare accuracies and time consumptions between two approaches for classifying vehicle color. The two approaches are ELM and X-ELM.

5.3.2 Experiment Setup

The experiment is set up as follows:

Dataset: Dataset from [18] is used for conducting experiments for vehicle color classification. All in all, 914 instances were collected. The dataset composes of 3 attributes, i.e. hue, saturation, and luminance in HSL color space, and 7 labeled classes, i.e. black, white, red, blue, yellow, green, and unknown. In addition, the 3 attributes are integer value ranged from 0 to 255. Furthermore, all evaluation models are trained with two third of the dataset and tested with another one third. For each run, instances for train and test are randomly selected based in mentioned ratio.

two third of the dataset is used as training data and other one third is used in testing

Attributes	Labeled Classes
Hue	Black
Saturation	White
Luminance	Red
	Blue
	Yellow
	Green
	Unknown

Table 5.6: List of attributes and labeled classes in vehicle color classification

Model	Testing accuracy (%)				Training time (ms)			
	Min	Mean	Max	SD	Min	Mean	Max	SD
ELM1	65.57	72.02	78.36	2.40	**11.24**	**13.85**	**18.67**	**1.21**
ELM2	70.82	75.46	80.33	2.20	13.47	17.09	46.27	3.62
ELM3	68.52	74.06	79.02	2.28	17.89	21.04	32.27	2.53
XELM1	**71.80**	**75.92**	**81.31**	**1.98**	70.67	79.37	89.80	5.10

Table 5.7: Comparison table of testing accuracy and training time of three ELM based models and one X-ELM based model for vehicle color classifying task

5.3.3 Evaluation Models

The details of evaluation models are described in Table 5.5. Evaluation models for vehicle color classification are set to be same as vehicle type classification. See subsection 5.2.3 for more details.

5.3.4 Results

The performances are measured in two terms which are same as measured in evaluating classifiers for a vehicle type. The outcome are shown in Table 5.7. Figure 5.3 and figure 5.3 represent Table 5.7 in graphical form. In addition, as the initial input weights are arbitrary generated, a hundred time of evaluations are performed to measure the average performance, peak performance, and the stability of a model. Four statistic values are shown, i.e. minimum, mean, maximum, and standard deviation.

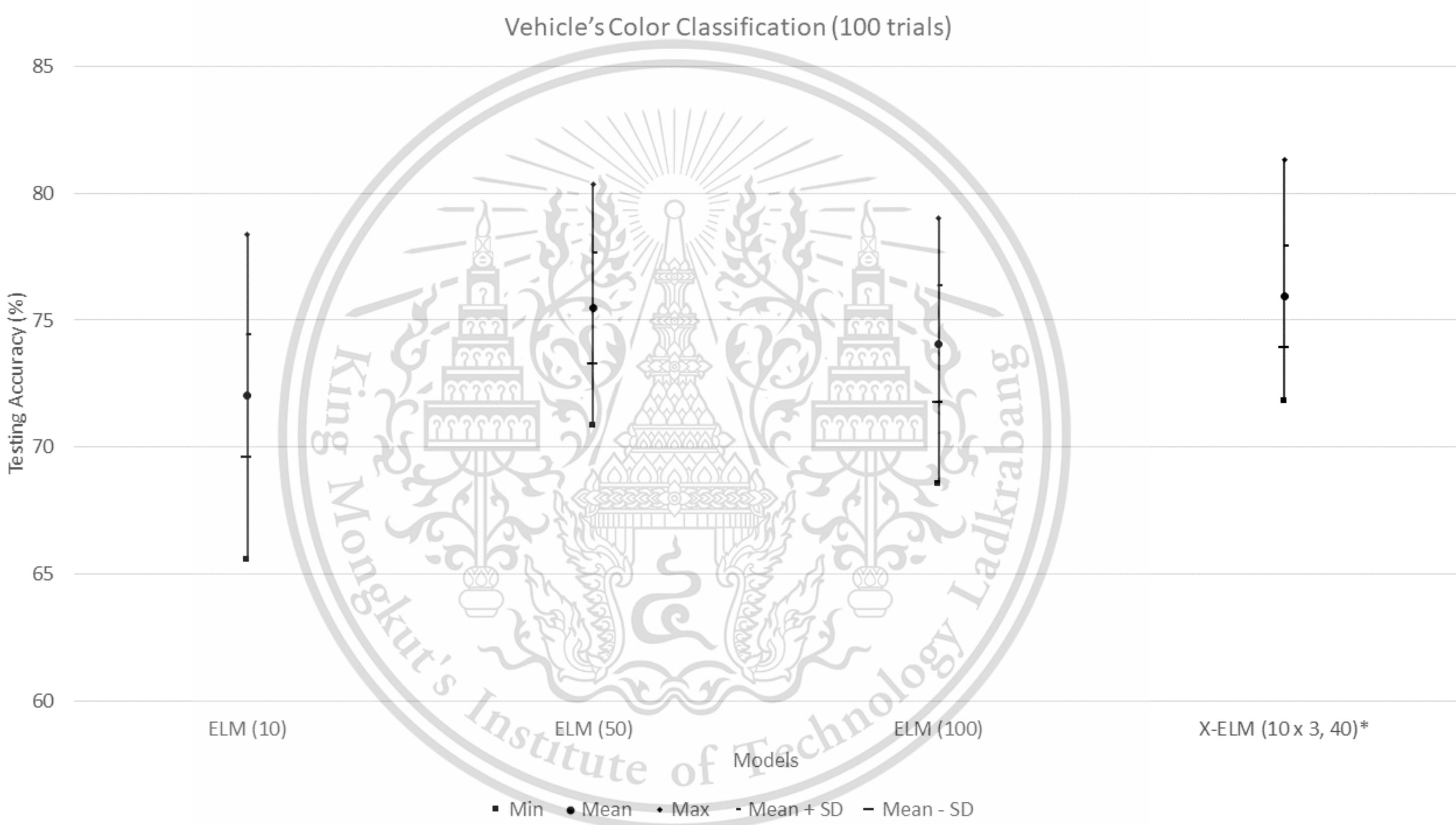

Figure 5.3: Comparison between models of all statistical measurements of testing accuracy on vehicle's color classification

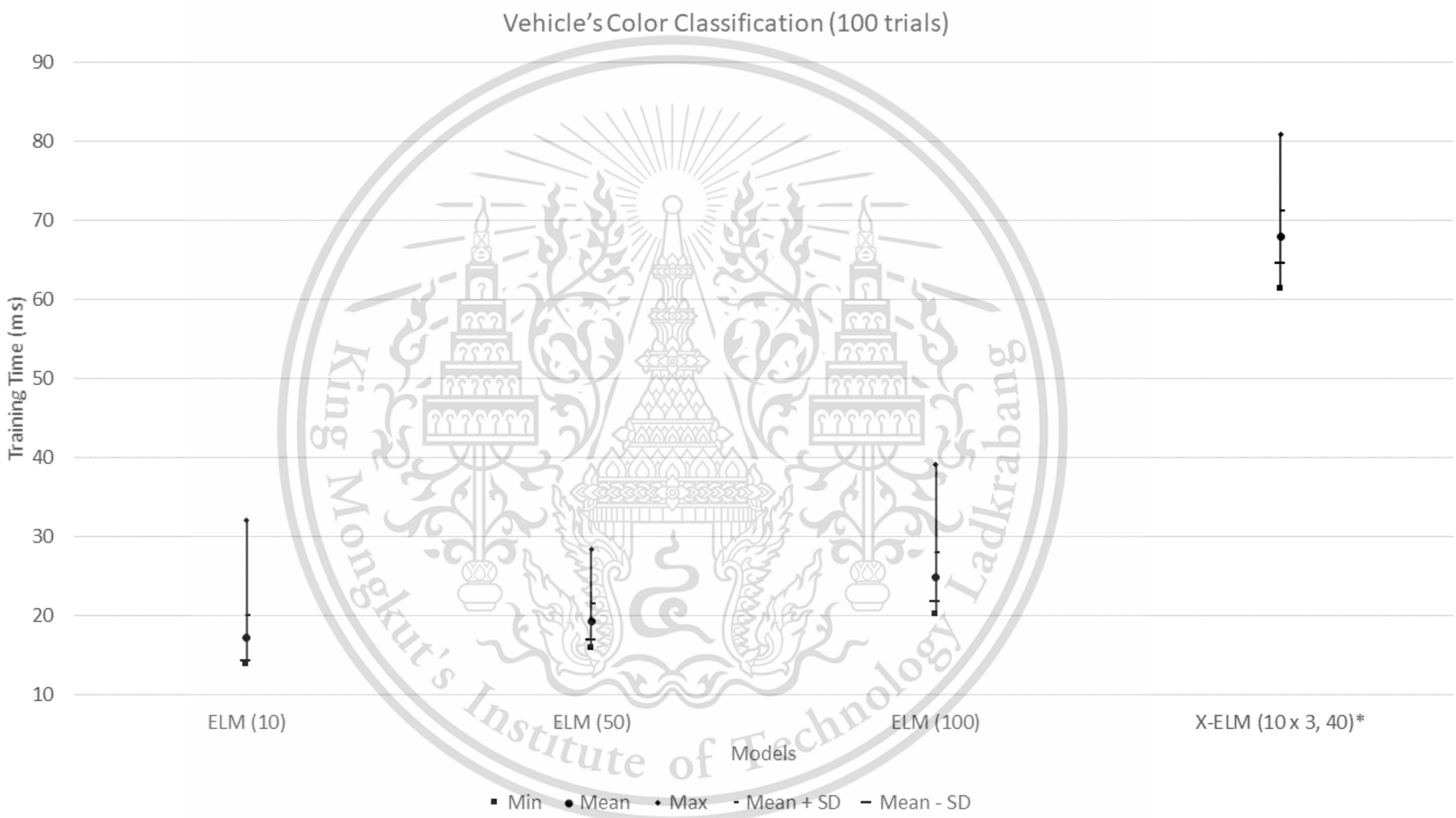

Figure 5.4: Comparison between models of all statistical measurements of training time on vehicle's color classification

5.3.5 Summary

This second experiment is conducted based on vehicle color classifying task. In this experiment, X-ELM also outperform ELM in all evaluated statistic measurements. ELM achieves testing accuracies of 70.82%, 75.46%, 80.33%, and 2.20% for minimum, mean, maximum, and standard deviation, respectively. X-ELM outperforms these with testing accuracies of 71.80%, 75.92%, 81.31%, and 1.98% for minimum, mean, maximum, and standard deviation, respectively. However, X-ELM consumes time more than ELM as it construct several ELM based network inside. All in all, X-ELM achieves better accuracies with lower of its variation.

5.4 Experiment 3: Modified National Institute of Standards and Technology Optical Character Recognition

5.4.1 Objective

This experiment is conducted to measure and compare accuracies and time consumptions between two approaches for recognizing or classifying hand written digits based on MNIST dataset. The two approaches are ELM and X-ELM.

5.4.2 Experiment Setup

The experiment is set up as follows:

Dataset: This experiment is conducted with dataset from MNIST. It is a dataset of hand written digits, from 0 to 9. In total, the dataset contain 60,000 instances of training data and 10,000 instances of testing data. Each of data instance composes of 28 by 28 pixels gray scale image as input features and 1 labeled classed of that image's digit as an output target. In addition, some sample from MNIST dataset is provided as shown in Figure 5.5. Moreover, list of attributes and labeled classed can also be viewed from Table 5.8.

Figure 5.5: Sample images from MNIST dataset

Attributes	Labeled Classes
28 by 28 pixels gray scale image	Label “0”
	Label “1”
	Label “2”
	Label “3”
	Label “4”
	Label “5”
	Label “6”
	Label “7”
	Label “8”
	Label “9”

Table 5.8: List of attributes and labeled classes in MNIST OCR

5.4.3 Evaluation Models

Models designed for this experiment are differed from the first two experiments. This is due to the task's complexity. In this experiment, 6 ELM based networks and 6 X-ELM based networks are evaluated. The details are shown in Table 5.9.

From Table 5.9, the first six models ELM4, ELM5, ELM6, ELM7, ELM8, and ELM9 have the same framework and activation function, which are ELM and sigmoid, respectively. However, number of hidden nodes are varied to observe effects on performance. The later six models XELM2, XELM3, XELM4, XELM5, XELM6, and XELM7 is constructed based on X-ELM. They are varied by number of predictors and number of hidden nodes. As MATLAB allows maximum array size of 30,256 megabytes, the maximum number of hidden neural node that can be created in this experiment is limited to 12,000.

5.4.4 Results

In this experiment, testing accuracy and training time are both focused and recorded. Testing accuracy is measured in term of percentage, range from 0 to 100, while training time is measured in term of seconds.

Model	Framework)	Architecture	Activation Function	Number of Hidden Nodes
ELM4	ELM	SLFN	Sigmoid	1,000
ELM5	ELM	SLFN	Sigmoid	2,000
ELM6	ELM	SLFN	Sigmoid	3,000
ELM7	ELM	SLFN	Sigmoid	5,000
ELM8	ELM	SLFN	Sigmoid	10,000
ELM9	ELM	SLFN	Sigmoid	12,000
XELM2	X-ELM	Multiple SLFNs	Sigmoid	(1,000 × 3, 200)*
XELM3	X-ELM	Multiple SLFNs	Sigmoid	(2,000 × 3, 200)*
XELM4	X-ELM	Multiple SLFNs	Sigmoid	(3,000 × 5, 400)*
XELM5	X-ELM	Multiple SLFNs	Sigmoid	(5,000 × 5, 400)*
XELM6	X-ELM	Multiple SLFNs	Sigmoid	(10,000 × 7, 600)*
XELM7	X-ELM	Multiple SLFNs	Sigmoid	(12,000 × 7, 600)*

* (M × P, R) There are P sub-classification models contained M neural nodes in their hidden layer. The extended network contains R neural nodes in its hidden layer.

Table 5.9: Information of models used in experiments and evaluation for Experiment 1 and Experiment 2

Due to limited resources, in this experiment, number of trials for each model run is designed to be ten. The results are shown in Table 5.10. First column is model names, whereas second column shown testing accuracies achieved for each model. The last column shows time consumed during the training of each model. Furthermore, the table is also represented in graphical form shown in Figure 5.6 and Figure 5.7. As there is limited array size, two models are highlighted out and shown in Figure 5.8. Note that graph showing training time in this experiment is displayed in logarithm scale.

With limited size of array, X-ELM allows to achieve better testing accuracy than ELM. However, it uses more time in training. The peak accuracy of 98.23% is achieved by X-ELM framework based network. The network consists of seven SLFN with 12,000-hidden nodes and another 600-hidden nodes SLFN as an extended part to combined those first seven SLFN. Total training time of 11,289.43 seconds is consumed.

Model	Testing accuracy (%)				Training time (s)			
	Min	Mean	Max	SD	Min	Mean	Max	SD
ELM4	93.30	93.52	93.74	0.14	15.36	15.64	15.82	0.16
ELM5	95.17	95.32	95.50	0.10	47.13	47.70	48.59	0.50
ELM6	95.89	96.08	96.32	0.13	101.59	105.66	110.46	3.14
ELM7	96.67	96.89	97.11	0.13	224.80	232.12	239.56	6.18
ELM8	97.39	97.57	97.73	0.11	977.71	1,020.76	1,060.07	29.67
ELM9	97.56	97.71	97.84	0.09	1,529.14	2,030.77	3,546.92	584.21
XELM2	94.87	95.02	95.13	0.10	51.47	52.01	52.64	0.40
XELM3	96.08	96.18	96.28	0.06	147.12	148.74	151.11	1.10
XELM4	96.90	96.97	97.10	0.07	525.93	538.29	553.12	10.78
XELM5	97.36	97.51	97.62	0.08	1,132.63	1,164.47	1,201.73	31.51
XELM6	98.06	98.11	98.18	**0.04**	6,876.46	7,365.97	9,539.34	851.60
XELM7	**98.16**	**98.22**	**98.30**	**0.04**	10,746.96	12,300.46	15,633.01	1,559.29

Table 5.10: Comparison table of testing accuracy and trainings of six ELM based models and six X-ELM based models for MNIST OCR task

Note that, in the experiment, non of multi-threading techniques is used. Only single thread coding is implemented.

Furthermore, Figure 5.6 illustrate that the trend of testing accuracy still increasing. As highlighted in Figure 5.6 and Figure 5.8, X-ELM highlighted in green shows that it could lower the variation of testing accuracy comparing to ELM highlighted in blue. Moreover, it also rises the accuracy too.

5.4.5 Summary

For this experiment, dataset of MNIST OCR is used. With a limited size of array, X-ELM achieves a maximum accuracies of 98.30%, while ELM achieves a maximum accuracies of 97.84%, Moreover, standard deviation of X-ELM is just 0.04, which is more than twice lower than of ELM. Similar to the first two experiments, the additional testing accuracy comes with trade off of training time.

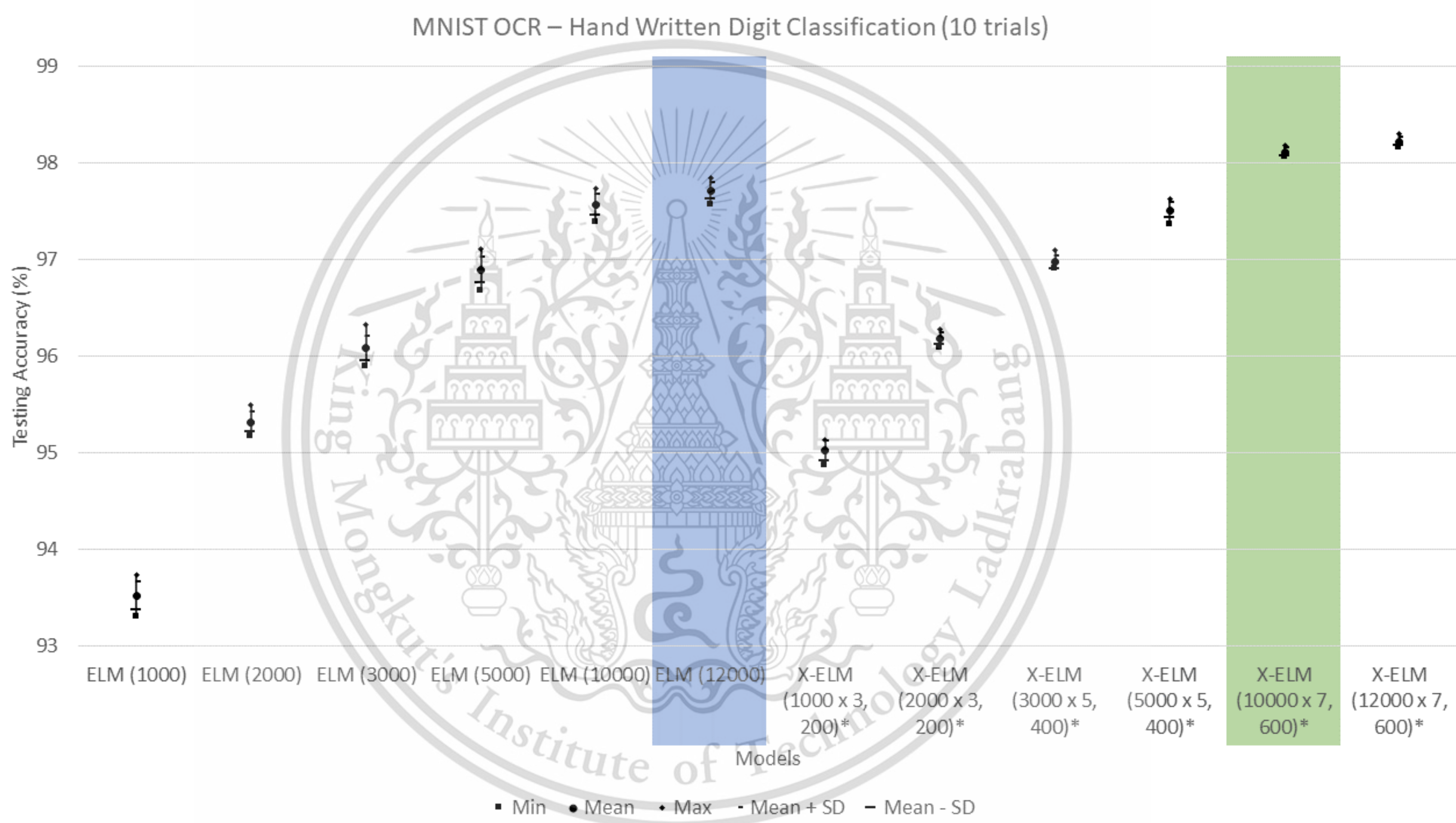

Figure 5.6: Comparison between models of all statistical measurements of testing accuracy on MNIST OCR

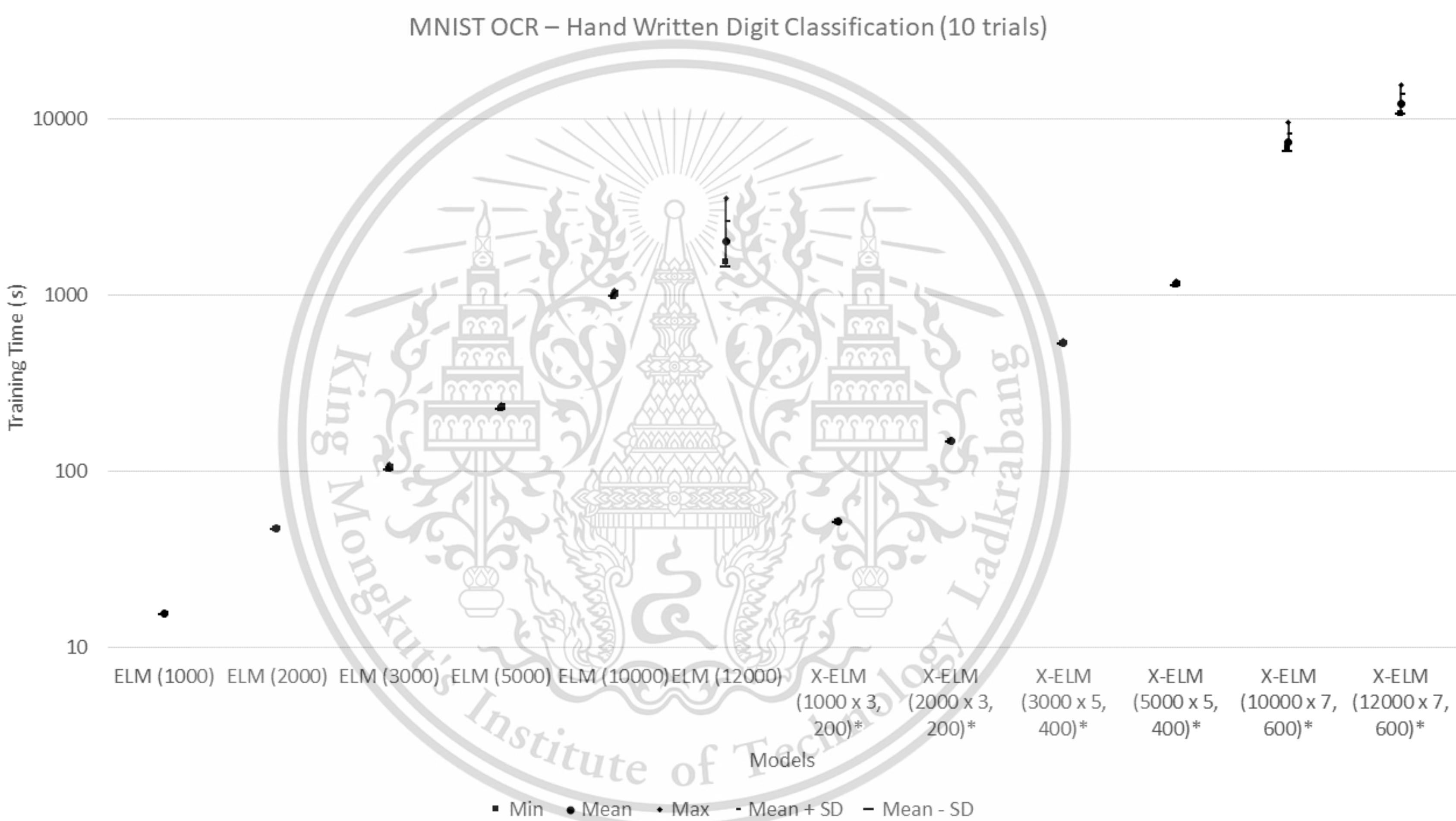

Figure 5.7: Comparison between models of all statistical measurements of training time on MNIST OCR

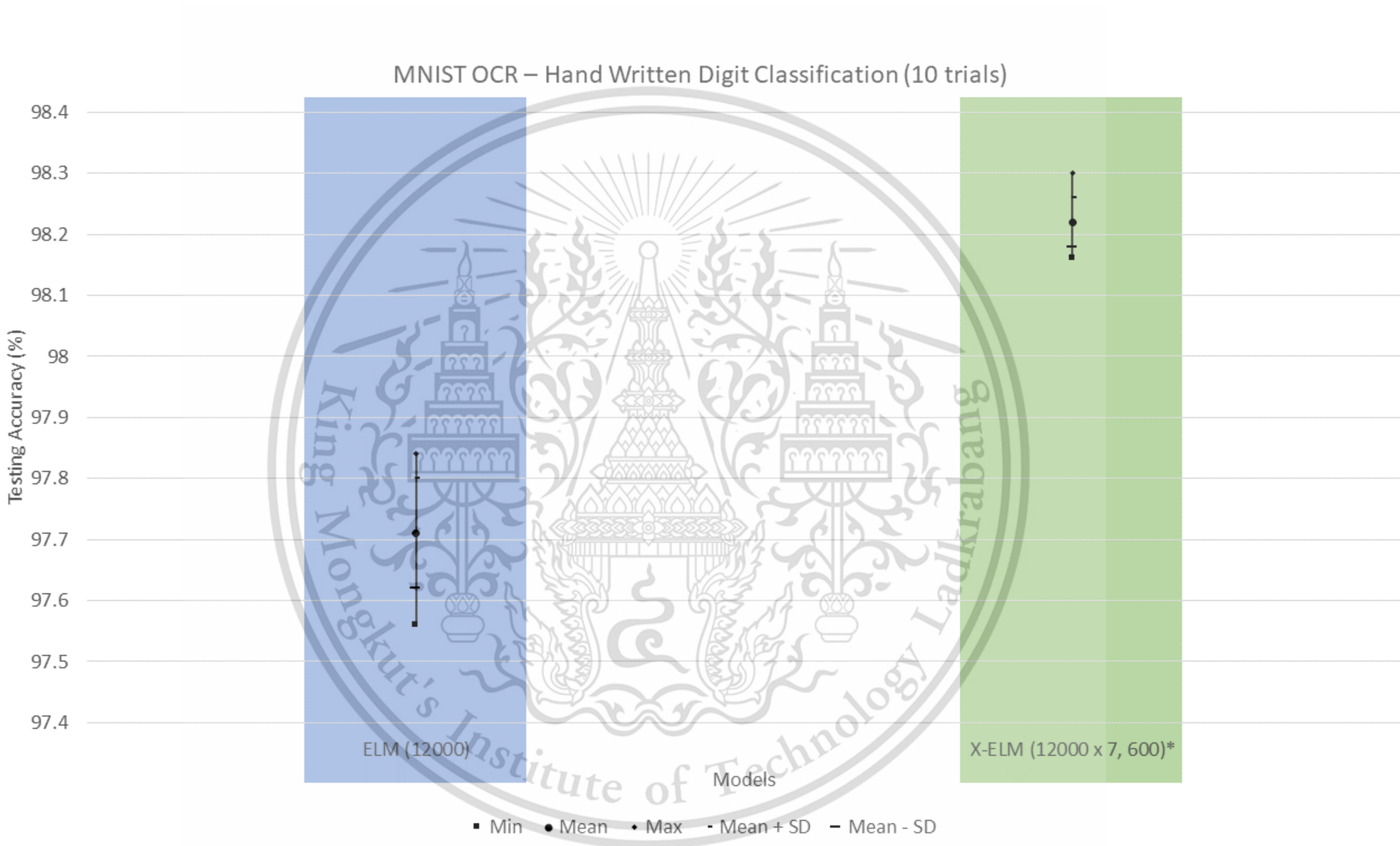

Figure 5.8: Comparison between two highlighted models of all statistical measurements of testing accuracy on MNIST OCR

In this chapter, the experiments are designed and conducted in order to evaluate performance of both ELM and X-ELM. The detailed summarize are described. In addition, the conclusion will be discussed in following chapter.

Chapter 6

Conclusion

This study is mainly focused on the proposed Extended Extreme Learning Machine (X-ELM). It is experimented with three classifying tasks, i.e. vehicle type, vehicle color, and Modified National Institute of Standards and Technology character classifications. The performances are then compared with the original Extreme Learning Machine (ELM).

In this study, there are three experiment's setups designed for each task. In every task, X-ELM could achiever better testing accuracy than original ELM. However, it takes more training time. Due to sequential training each predictor in X-ELM, it does not consume more memory than ELM.

X-ELM shows possibility of more complexity network comparing to ELM. ELM can only used to train SLFN, while X-ELM can stack networks into a deeper network. Moreover, with limited resources, e.g. memory, it can achieve higher accuracy. In conclusion, X-ELM gives more overall consistent accuracy in term of both higher average testing accuracy and lower standard deviation. Moreover, with limited amount of memory usage, X-ELM shows a possibility of getting a higher testing accuracy. These advantages come with the trade-off in amount of training time consumed and slightly more parameters tuning.

Chapter 7

Discussion

Although the proposed framework seem to work very well in many applications, it can be further improved in several model aspects. The neural based X-ELM model can be stacked to create deeper networks, e.g. one X-ELM inside another X-ELM. Furthermore, each predictive model inside can be different parameters, e.g. different number of hidden nodes.

As the experiments are all done in sequential training. If parallel processing is applied, more resources of memory may be needed because each predictor in X-ELM required its own resources in training.

The author also perform minor experiments in many other datasets, i.e., iris flower dataset, ecoli dataset, and forest fires dataset, as well. For these dataset, the author did not measure the stability. However, from the author's observation, X-ELM could perform and achieve higher peak accuracy than ELM can. Nevertheless, the model can be improved by varying predictive models, such as including Random Forest or Support Vector Machine.

References

[1] A. Zaknich and Y. Attikiouzel, "Detection of sodium oxalate needles in optical images using neural network classifiers," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, Nov 1995, pp. 1699–1702 vol.4.

[2] F. Xin, S. Ni, H. Li, and X. Zhou, "General regression neural network and artificial-bee-colony based general regression neural network approaches to the number of end-of-life vehicles in china," *IEEE Access*, vol. 6, pp. 19 278–19 286, 2018.

[3] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, July 2006.

[4] Y. Bengio, *Learning Deep Architectures for AI*. Now Foundations and Trends, 2009. [Online]. Available: https://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=8187120

[5] S. Linnainmaa, "Taylor expansion of the accumulated rounding error," *BIT Numerical Mathematics*, vol. 16, no. 2, pp. 146–160, Jun 1976.

[6] L. Tienan, R. Weijian, C. Guangyi, X. Baochang, and Y. Di, "Second order back-propagation learning algorithm and its application for neural network," in *Proceedings of the 3rd World Congress on Intelligent Control and Automation (Cat. No.00EX393)*, vol. 2, 2000, pp. 817–820 vol.2.

[7] T. Paine, H. Jin, J. Yang, Z. Lin, and T. S. Huang, "GPU asynchronous stochastic gradient descent to speed up neural network training," *CoRR*, vol. abs/1312.6186, 2013. [Online]. Available: http://arxiv.org/abs/1312.6186

[8] J. Keuper, “Distributed training of deep neuronal networks: Theoretical and practical limits of parallel scalability,” *CoRR*, vol. abs/1609.06870, 2016.

[9] G. B. Huang, Q. Y. Zhu, and C. K. Siew, “Extreme learning machine: a new learning scheme of feedforward neural networks,” in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, vol. 2, July 2004, pp. 985–990 vol.2.

[10] P. L. Bartlett, “The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network,” *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 525–536, Mar 1998.

[11] J. Cao, Y. Zhao, X. Lai, M. E. H. Ong, C. Yin, Z. X. Koh, and N. Liu, “Landmark recognition with sparse representation classification and extreme learning machine,” *Journal of the Franklin Institute*, vol. 352, no. 10, pp. 4528–4545, 2015.

[12] P. D. Moerland and E. Fiesler, “Handbook of neural computation: E1.2. neural network adaptations to hardware implementations,” in *Proceedings of the fourth symposium on Natural Language Processing*. New York: Institute of Physics Publishing and Oxford University Publishing, 1996, pp. 1–13.

[13] G. B. Huang, Q. Y. Zhu, and C. Siew, “Extreme learning machine: A new learning scheme of feedforward neural networks,” in *Proceedings of International Joint Conference on Neural Networks 2014*, Jul. 2004.

[14] W. Zhu, J. Miao, L. Qing, and G. B. Huang, “Hierarchical extreme learning machine for unsupervised representation learning,” in *2015 International Joint Conference on Neural Networks (IJCNN)*, July 2015, pp. 1–8.

[15] Z. Huang, Y. Yu, and J. Gu, “A novel method for traffic sign recognition based on extreme learning machine,” in *Proceeding of the 11th World Congress on Intelligent Control and Automation*, June 2014, pp. 1451–1456.

[16] M. Ishihara and Y. Ishihara, “A pilot study on impact of viewing distance to task performance,” in *2017 International Conference on Biometrics and Kansei Engineering (ICBAKE)*, Sept 2017, pp. 139–143.

[17] J. Chang, L. Wang, G. Meng, S. Xiang, and C. Pan, "Vision-based occlusion handling and vehicle classification for traffic surveillance systems," *IEEE Intelligent Transportation Systems Magazine*, vol. 10, no. 2, pp. 80–92, Summer 2018.

[18] K. Saripan, C. Deachakul, and W. Moungmai, "Smart vehicle search in surveillance video," Bachelor Thesis, King Mongkut's Institute of Technology Ladkrabang, 2015.

[19] G. B. Huang, Q. Y. Zhu, and C. K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006, neural Networks.

[20] P. L. Bartlett, "The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network," *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 525–536, Mar 1998.

[21] scikit-learn developers. (1999) sklearn.preprocessing.onehotencoder. [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html

Application of Neural Networks for Vehicle Classifiers: Extreme Learning Machine Approach

Boonnithi Jiaramaneepinit
International College
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand
60610021@kmitl.ac.th

Chaiwat Nuthong
International College
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand
chaiwat.nu@kmitl.ac.th

***Abstract*—Machine learning has been a popular topic in research field for many applications. One of the applications is traffic surveillance system. In many areas, traffic surveillance system is installed in order to gather and estimate important traffic information. Nowadays, there are several systems used for information's extracting, and classifying. One of the well-known approaches is decision tree. It uses a tree-like model that decides consequences outcomes from events. However, in some application, decision tree does not perform well. Another widely used approach is neural network, which has promising performance. It has been developed and become one of the most popular computing systems in research field. The traditional approach in training neural network is backpropagation. However, it has several drawbacks. One of them is the training time. In recent decades, Extreme learning machine (ELM) was proposed for training single hidden layer feed-forward neural network (SLFN) in the extremely fast way. It minimizes training error by utilizing dataset in one-shot calculation. This paper focuses on classifiers in traffic surveillance system. The classification divides into two main tasks. One is vehicle types' classification. Another is vehicle colors' classification. Neural networks trained with ELM are applied to the dataset. The performance are then compared to decision tree based approaches with ensemble methods. The experimental results show that ELM achieves better accuracy than of decision tree based approaches in both tasks.**

***Index Terms*—machine learning, classification, decision tree, ensemble method, neural network, single hidden layer feed-forward neural network, extreme learning machine, traffic surveillance system**

I. Introduction

Nowadays, several traffic surveillance cameras are installed in many areas. Their objectives is to monitor traffic and circumstances. They are used in several applications. One of them is an event searching. In many incidents, police officers spend a lot of time and effort in searching suspected objects or vehicles manually using surveillance camera monitors. However, spending a long time focusing in front of monitors leads to eye fatigue. Moreover, manually searching in an inappropriate environment leads to low searching accuracy [1]. For these reasons, several machine learning approaches are implemented and applied to surveillance system.

There are many proposed systems in recent decades. Decision tree is used for vehicle classification by K. Ying et al. and K. Saripan et al. [2], [3]. Moreover, S. B. Changalasetty et al. applied multi-layer perceptron for classifying vehicles as big or small using vehicle features from traffic video [4]. Z. Zhang used deep neural network to classify road vehicle [5]. However, many classifiers required a lot of resources and training time. For this reasons, authors would like to propose systems for vehicle classifications with high performance and low training time.

This paper proposes a system to classify vehicles using their extracted characteristics from traffic surveillance systems. Neural networks trained using Extreme Learning Machine (ELM) are applied in order to acquire well-performed systems with reasonable training time. In addition, the experiments focuses on applying neural networks with ELM. The training time and accuracies of the aforementioned systems are then measured and compared with tree-based classifiers.

The remainder of this paper is organized as follows. Section II explains a preliminary of decision tree with introductory concept of ensemble method. This is followed by Section III which discusses overall idea of single hidden layer feed-forward neural network and its training method used in this paper. Section IV explains environmental setup, environment, and models applied in order to conduct experiments for evaluation. Finally, Section V concludes all experiments and is then followed by discussion and future works.

II. Decision Tree

Decision tree is a predictive model in machine learning based on tree structure. It is one of supervised machine learning approach based on divide and conquer strategy. In addition, it uses an inductive way of learning and specific information to create generalized conclusion [6].

Decision tree consists of two main elements, i.e. nodes and branches. Branches or edges connect from node to the following node ans is corresponded to a value from the test. Nodes can be categorized into two different types. First type is a decision node, also known as a non-leaf node. The input to this type of nodes is tested based on the a value of certain attribute and the condition. The outcome is then to follow that branch to the next node. Nevertheless, the process is repeat iteratively. Second type of nodes is end nodes or leaf nodes. This type of nodes is where the process ended. Furthermore, it shows output target for the input instance.

There are several approaches in building decision tree. ID3 or Iterative Dichotomiser 3 is an algorithm for decision tree learning proposed by R. Quinlan [7]. However, the algorithm has been extended and superseded by C4.5 algorithm [8]. C4.5 algorithm applied information theory in order to acquire a value of information gain on each feature. The values are then compared and chosen. The process is repeated until the data become homogeneity or the requirements is met.

A. Ensemble Method

Ensemble method is a learning approach that use multiple learning algorithms or models. Its objective is to build a strong classifier that has good predictive performance using weak classifier. One of the powerful well-known ensemble methods is Adaptive Boosting (AdaBoost). It was proposed by Y. Freund and R. E. Schapire [9]. Furthermore, AdaBoost combines outputs from weak learners to decide the final outcome using weighted sum. Another well-known ensemble method is random decision forest or random forest [10]. Random forest applies several decision trees as its weak classifiers. These decision trees are trained with partial dataset from the whole dataset. The final outcome using random forest approach is decided by statistical information, which is the same as bootstrap aggregating method [11].

III. Single Hidden Layer Feed-forward Neural Network

Single-hidden layer feedforward neural network (SLFN) is one of artificial neural networks, which inspired from neural from human brain [12]. Its network is directed and has no cycle. Moreover, SLFN consists of two main compositions, i.e. processing units and weighted links. Processing units are also known as hidden nodes or artificial neurons. Furthermore, the input features are sent to processing units with weighted links as a signal. The output of each unit is computed by a non-linear function of the summation of its input signal. This function is also know as activation function.

The conventional approach in training SLFN is backward propagation of errors or backpropagation. It adjusts the weights that link between neurons by feeding signal in and calculating for error. In addition, it is an iterative approach which based on gradient descent [13]. Although using backpropagation has promising performance, it has several drawbacks. Training neural network using backpropagation requires huge amount of time and resource. It could take from several hours to a day or even a week [14], [15]. For this reason, several optimizations for backpropagation are being researched [16]–[18].

According to G. B. Huang et al. [19], for N distinct samples $(\mathbf{x}_i, \mathbf{t}_i)$, where $\mathbf{x}_i$ is an input instances vector $[x_{i1}, x_{i2}, ..., x_{in}]^T \in \mathbf{R}^n$ and $\mathbf{t}_i$ is an expected output vector $[t_{i1}, t_{i2}, ..., t_{im}]^T \in \mathbf{R}^m$, the outputs from the network with $\tilde{N}$ hidden nodes can be written as

$$\mathbf{y}_j = \sum_{i=1}^{\tilde{N}} \boldsymbol{\beta}_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i), \quad \text{for} \quad j = 1, ..., N \tag{1}$$

where $\mathbf{w}_i$ is a weight vector $[w_{i1}, w_{i2}, ..., w_{in}]^T$ from input nodes to i^{th} hidden node, called input weights, $\boldsymbol{\beta}_i$ is a weight vector $[\beta_{i1}, \beta_{i2}, ..., \beta_{im}]^T$ from the i^{th} hidden node to output nodes, called output weights, b_i is a bias of the i^{th} hidden node, and $g(\cdot)$ is an activation function.

In the training phase, $\mathbf{y}_j$ is substituted by the expected output vector $\mathbf{t}_j$ to approximate N samples' outputs with minimized error. This can be written as

$$\mathbf{t}_j = \sum_{i=1}^{\tilde{N}} \boldsymbol{\beta}_i g(\mathbf{w}_i \cdot \mathbf{x}_j + b_i), \quad \text{for} \quad j = 1, ..., N \tag{2}$$

For compactible representation, equation (2) can be rewritten into

$$\mathbf{T} = \mathbf{H}\boldsymbol{\beta} \tag{3}$$

where $\mathbf{H}$ is an output matrix from the hidden layer with input $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N$. The i^{th} column of $\mathbf{H}$ is corresponded to the output from i^{th} hidden node. It can be formulated as

$$\mathbf{H} = \begin{bmatrix} g(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \dots & g(\mathbf{w}_{\tilde{N}} \cdot \mathbf{x}_1 + b_{\tilde{N}}) \\ \vdots & \ddots & \vdots \\ g(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \dots & g(\mathbf{w}_{\tilde{N}} \cdot \mathbf{x}_N + b_{\tilde{N}}) \end{bmatrix}.$$

$\boldsymbol{\beta}$ is an output weights matrix, while $\mathbf{T}$ is a target matrix. Matrix $\boldsymbol{\beta}$ and $\mathbf{T}$ can be expressed as

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_1^T \\ \vdots \\ \boldsymbol{\beta}_{\tilde{N}}^T \end{bmatrix}, \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{bmatrix}$$

respectively.

A. Extreme Learning Machine

Extreme Learning Machine (ELM) was neural network training approach proposed by G. B. Huang et al. [19]. ELM is claimed to have promising performance with extremely fast training time [20]. It solves two main drawbacks from backpropagation. The first drawback is the slow training speed. Instead of iteratively fix error based on gradient descent, ELM computes for necessary weight parameters using Moore-Penrose generalized inverse [21]. The second drawback is an iterative parameters tuning of the model.

ELM calculates output weights $\boldsymbol{\beta}$ from equation (3) by applying Moore-Penrose generalized inverse. To find the aforementioned weights $\boldsymbol{\beta}$, the assigned values of parameters $\mathbf{w}_i$ and b_i are randomly generated. Based on Moore-Penrose generalized inverse and a minimum-norm least squares, the matrix $\boldsymbol{\beta}$ can then be solved by

$$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T} \tag{4}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$. This allows the result's error to be minimized and the norm of weights become smallest in training process [20].

TABLE I
LIST OF INPUT FEATURES AND CLASSES IN VEHICLE TYPE CLASSIFYING TASK

Input Features	Classes
Characteristic of bounding box • X coordinate from top-left corner • Y coordinate from top-left corner • Height • Width Area's ratio of vehicle over background	Small Medium Large Unknown

TABLE II
LIST OF INPUT FEATURES AND CLASSES IN VEHICLE COLOR CLASSIFYING TASK

Input Features	Classes
Values of color in HSL color space • Hue • Saturation • Luminance	Black White Red Blue Yellow Green Unknown

IV. EXPERIMENTS

A. Datasets

The experiments are conducted using vehicles classifications dataset [3], [22]. The datasets collected 914 samples of vehicles and their characteristics with their class. The dataset are separated into two tasks, i.e. type and color classifying tasks. Lists of input features and target classes of type and color classification are shown in TABLE I and TABLE II, respectively.

In training phase, 75% of dataset is used for training models. The other 25% of dataset is split for testing. Two characteristics, i.e. bounding box of a vehicle and a ratio of vehicle over background, are used for vehicle type classification, while color's values in HSL color space are used for vehicle color classification [3].

B. Environments

All processes and data preparation are conducted in MATLAB 2017b under Windows 10 Pro 64-bit (10.0, Build 16299). The desktop computer is equipped with the following specifications:

- Intel Core i5-7200U CPU running at 2.50 GHz with 4 CPU threads
- 8192 MB of DDR4 memory running at 2133 MHz
- Solid state drive with capacity of 256 GB

C. Evaluation Models

There are seven models constructed with different methods and parameters in order to compare their performance. Details of the models are shown in TABLE III. The first three models are based on tree structure, i.e. decision tree, AdaBoost, and random forest. For the later four models, neural network trained with ELM are constructed by varying number of hidden node. The numbers of hidden nodes are 10, 20, 50, and 100, respectively. These are increased by investigating on the accuracy.

TABLE III
DETAILS OF MODELS USED IN CONDUCTING EXPERIMENTS

Model	Method	Activation Function	Number of Hidden Nodes
DT	Decision tree	-	-
AB	AdaBoost	-	-
RF	Random forest	-	-
NN1	Neural network (ELM)	Sigmoid	10
NN2	Neural network (ELM)	Sigmoid	20
NN3	Neural network (ELM)	Sigmoid	50
NN4	Neural network (ELM)	Sigmoid	100

Remark Model DT, AB, and RF are constructed same as [22]

For models NN1, NN2, NN3 and NN4, a hundred times of training trials are run in order to collect and measure for statistical information. This is due to the arbitrary The performance of each model is then evaluated. Moreover, the experiments computes for minimum value, average value, maximum value of training time and testing accuracy. Training time and testing accuracies of each model are compared and discussed in following subsection.

D. Experimental Results

From the experiments, conventional ELMs achieve peak accuracies of 89.96% and 83.41% for vehicle type and color classification, respectively. TABLE IV shows testing accuracies and training time achieved from each model for vehicle type classifying task, while TABLE V shows testing accuracies and training time of all models in conducting vehicle color classification.

For vehicle type classification, model NN4 achieves the best testing accuracy for all measurements, including minimum, average, and maximum values. Moreover, comparing model NN4 to models AB and RF, the average training time of model NN4 is lower than of models AB and RF.

For vehicle color classification, model NN3 achieves the best testing performance on all accuracies' measurements. In addition, training time of model NN3 is four time lower comparing to models AB and RF.

V. CONCLUSION AND DISCUSSION

In this paper, a novel training approach for neural networks, called Extreme Learning Machine, is applied for vehicle classifications based on the characteristics. The experiments compare results between tree-based classifiers and neural networks trained with ELM. They shows that neural networks with ELM have better testing performances with low average

TABLE IV
COMPARISON OF TESTING ACCURACIES AND TRAINING TIME OF ALL MODELS FOR VEHICLE TYPE CLASSIFICATION

Model	Testing accuracy (%)			Training Time (s)		
	Min	Mean	Max	Min	Mean	Max
DT	79.38			**<0.01**		
AB	78.94			0.04		
RF	79.82			0.10		
NN1	60.26	75.73	84.72	**<0.01**	0.02	0.03
NN2	70.74	80.90	88.21	0.01	0.02	0.03
NN3	74.67	82.68	89.09	0.01	0.02	0.04
NN4	**76.86**	**84.03**	**89.96**	0.02	0.02	0.04

Remark Minimum, average, and maximum values are computed for a thousand trial runs of training neural network using Extreme Learning Machine.

TABLE V
COMPARISON OF TESTING ACCURACIES AND TRAINING OF ALL MODELS FOR VEHICLE COLOR CLASSIFICATION

Model	Testing accuracy (%)			Training Time (s)		
	Min	Mean	Max	Min	Mean	Max
DT	67.98			**0.01**		
AB	69.29			0.08		
RF	68.42			0.11		
NN1	59.39	71.62	79.91	**0.01**	0.02	0.03
NN2	66.38	74.82	82.97	**0.01**	0.02	0.04
NN3	**68.12**	**75.83**	**83.41**	**0.01**	0.02	0.04
NN4	67.69	74.37	81.66	0.02	0.03	0.05

Remark Minimum, average, and maximum values are computed for a thousand trial runs of training neural network using Extreme Learning Machine.

training time of around 0.02 seconds. As the data contains nonlinearity, neural network, which able to handle nonlinear data, performs better than decision tree based models. Nevertheless, the authors aim to improve the testing accuracies by applying deep neural network architecture to the tasks. Moreover, as ELM are generalized for any classification task, the authors also intend to apply it to more tasks with other neural network architecture.

ACKNOWLEDGMENT

The authors would like to give our deep gratitude to Applied Machine Learning and Computer Vision laboratory at the International College, King Mongkut's Institute of Technology Ladkrabang for providing us supports. The authors also would like to thank Kiatkachorn Saripan for advices during the research. Special thanks to Hinagiku Katsura for the encouragement. Lastly, our grateful thanks are also extended to all, who directly or indirectly, have a part in this venture.

REFERENCES

[1] M. Ishihara and Y. Ishihara, "A pilot study on impact of viewing distance to task performance," in *2017 International Conference on Biometrics and Kansei Engineering (ICBAKE)*, Sept 2017, pp. 139–143.

[2] K. Ying, A. Ameri, A. Trivedi, D. Ravindra, D. Patel, and M. Mozumdar, "Decision tree-based machine learning algorithm for in-node vehicle classification," in *2015 IEEE Green Energy and Systems Conference (IGESC)*, Nov 2015, pp. 71–76.

[3] K. Saripan, C. Deachakul, and W. Moungmai, "Smart vehicle search in surveillance video," 2015.

[4] S. B. Changalasetty, A. S. Badawy, L. S. Thota, and W. Ghribi, "Moving vehicles classification in weka," in *2015 International Conference on Advanced Computing and Communication Systems*, Jan 2015, pp. 1–6.

[5] Z. Zhang, C. Xu, and W. Feng, "Road vehicle detection and classification based on deep neural network," in *2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, Aug 2016, pp. 675–678.

[6] E. Alpaydin, *Introduction to Machine Learning*, 2nd ed. The MIT Press, 2010.

[7] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, Mar. 1986.

[8] ——, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[9] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[10] T. K. Ho, "Random decision forests," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, Aug 1995, pp. 278–282 vol.1.

[11] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, Aug. 1996.

[12] D. Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. New York, NY, USA: Henry Holt and Co., Inc., 1982.

[13] S. Linnainmaa, "Taylor expansion of the accumulated rounding error," *BIT Numerical Mathematics*, vol. 16, no. 2, pp. 146–160, Jun 1976.

[14] F. N. Iandola, K. Ashraf, M. W. Moskewicz, and K. Keutzer, "Firecaffe: near-linear acceleration of deep neural network training on compute clusters," *CoRR*, vol. abs/1511.00175, 2015.

[15] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1912–1920.

[16] O. N. A. AL-Allaf, "Fast backpropagation neural network algorithm for reducing convergence time of bpnn image compression," in *ICIMU 2011 : Proceedings of the 5th international Conference on Information Technology Multimedia*, Nov 2011, pp. 1–6.

[17] H. A. Abbass, "Speeding up backpropagation using multiobjective evolutionary algorithms," *Neural Computation*, vol. 15, no. 11, pp. 2705–2726, Nov 2003.

[18] M. E. Rimer, T. L. Andersen, and T. R. Martinez, "Speed training: improving the rate of backpropagation learning through stochastic sample presentation," in *Neural Networks, 2001. Proceedings. IJCNN '01. International Joint Conference on*, vol. 4, 2001, pp. 2661–2666 vol.4.

[19] G. B. Huang, Q. Y. Zhu, and C. K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, vol. 2, July 2004, pp. 985–990 vol.2.

[20] ——, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006, neural Networks.

[21] C. R. Rao and S. K. Mitra, *Generalized Inverse of Matrices and its Applications*. Wiley, 1971.

[22] K. Saripan and C. Nuthong, "Tree-based vehicle classification system," in *2017 14th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, June 2017, pp. 439–442.