SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
ISSN 2054-7390

Detection of the Onset of Diabetes Mellitus by Bayesian Classifier Based Medical Expert System

[1]Md. Mozaharul Mottalib, [2]Md. Mokhlesur Rahman, [3]Md. Tarek Habib and [4]Farruk Ahmed
[1]Department of Computer Science and Engineering, Green University, Bangladesh;
[2]Department of Computer Science and Engineering, Prime University, Bangladesh;
[3]Department of Computer Science and Engineering, Daffodil International University, Bangladesh;
[4]Department of Computer Science and Engineering, Independent University, Bangladesh
mm.mottalib@gmail.com; mmarks_cse@yahoo.com; md.tarekhabib@yahoo.com; farruk60@gmail.com

ABSTRACT

Expert systems play an important role in medical diagnosis research. Researches are still being conducted for building expert systems capable of diagnosing different diseases. Diabetes mellitus is one of the diseases that have gained attention in the past years. Patients are usually unaware of having this disease and are finally diagnosed with diabetes after several years from onset. Since diabetes can be controlled, it is much desirable to harness it at the onset. Therefore, the prediction of onset of diseases like diabetes has been the point of interest for the researchers. Researchers are continuously trying to formulate an inference engine, a part of an expert system, in order to predict the disease at the beginning. In this paper, we present a Bayesian classification approach to identify the onset of diabetes mellitus in patients using a well-known data set as the sample. We have found an intriguing result with more than 87% accuracy.

Keywords: Expert system, diagnosis of disease, pattern recognition, classification, Bayesian classifier.

1 Introduction

Expert systems are the special type of systems, which offer the solution of different problems through providing suggestions equivalent to human experts in those particular fields [1]. Real-life problems are solved using expert systems, specifically the problems that do not have predefined solution in general. Expert systems are used in the fields where sufficient amount of human expertise is required. Examples are the medical diagnosis of disease, financial advice, designing of products, etc. In the sector like the medical diagnosis of diseases, the inference engine of an expert system is built following several techniques. Pattern recognition is one of the well-known techniques.

Pattern recognition and data mining are used in different fields of our life. Especially, these techniques are more frequent in military, medical and industrial areas. With the passage of time, data collection and analysis have been drastically improved. New technologies contributed in these aspects. Possibilities of new researches are dramatically increased.

Since data analysis has traversed a lot of advancements and is continuously experiencing more, analysis of diseases in medical sectors got growth as well. Researchers have been doing research on how to predict the onset of diseases before any harm occurs or how to minimize the adversities.

DOI: 10.14738/tmlai.44.1962
Publication Date: 19th July, 2016
URL: http://dx.doi.org/10.14738/tmlai.44.1962

However, diabetes is one of the diseases those are under-diagnosed [2]. About one-thirds of the patients with diabetes are not aware of their having it. An average of 7years is the period between onset and diagnosis. Diabetes is a disease that needs constant monitoring and it could substantially decrease the life quality. As other diseases, early diagnosis of diabetes is crucial and can reduce the harm inflicted on the body.

We selected the Pima Indians Diabetes Dataset, available in [3], for building our model to predict the onset of diabetes mellitus. The description of the data set is described later.

We arranged the rest of the paper as follows. Section II discusses the related works in this field. Section III is used for describing the methodology of our classification model. Section IV contains the implementations. Experimental result and comparative analysis are given in the section V and finally, we conclude our work by giving future direction in section VI.

2 Literature Review

As it has been noted that researchers are continuously working on predicting diseases at onset, there have been several works in this field. Some works are related to the diagnosis of diseases while many are done on diabetes itself.

A method of biomedical signal classification using complex-valued pseudo autoregressive (CAR) modelling approach was proposed in [4]. An improvement on traditional multilayer perceptron (MLP) has been proposed in [5]. Solely on diabetes, work has been done using principal component analysis (PCA) and adaptive neuro-fuzzy adaptive systems in [6]. Generalized discriminant analysis (GDA) and least square support vector machine (LS-SVM) was used and a new cascade learning system based on that GDA and LS-SVM was proposed in [7].H. Kahramanli et al. presented a hybrid neural network that includes artificial neural network (ANN) and fuzzy neural network (FNN) [8].

Jack W. Smith et al. used ADAP learning algorithm to discern the onset of diabetes mellitus [9]. Neural network is also used for diabetes mellitus prediction in [10]. Again, Bayesian network was implemented for prediction of type-2 diabetes in [11]. Besides these, there are more works on disease classification and also on diabetes prediction. We only discuss the works that closely resembles our work.

3 Methodology

Our approach starts with building a model. After building the model, we train the model with the help of training dataset. Then applying the model in test samples we retrieve some predictions. These predictions tell us whether a patient or sample is in risk of diabetes mellitus foreknowing the onset. Figure 1 depicts the total overview.

3.1 Expert Systems

Expert systems consist two essential components: knowledge base and inference engine. A knowledge base is a repository for the domain-relevant knowledge. Algorithms for handling the knowledge base are the main attributes of an inference engine. A specific expert system is developed using resources from various knowledge banks, such as human experts, textbooks, and databases. In our case, the source is Pima Indian Database.

The knowledge base is subject to changes. Since our concern is not the knowledge base but the inference engine, we built the knowledge data set from the above mentioned source.

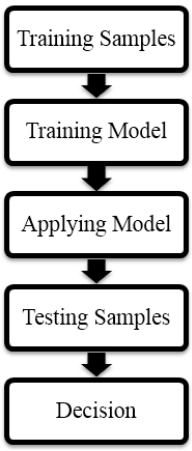


Figure 1. The approach for building the model for predicting the onset of diabetes mellitus.

Figure 2 demonstrates the simple architecture of a global expert system. We are focusing on the inference part. The inference engine we built follows the Bayesian theorem for predicting disease, in this case, diabetes mellitus.

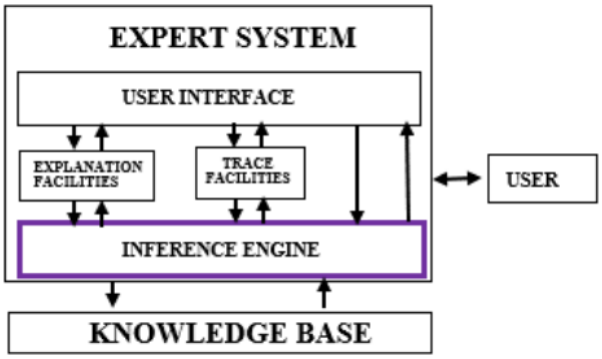


Figure 2. Global architecture of an expert system.

3.2 Dataset Description

The Pima Indian population who resides near Phoenix, Arizona was the research population. The National Institute of Diabetes, Digestive, and Kidney Diseases has been constantly studying the population since 1965 because of its high incidence rate of diabetes [12]. Community residents over the age of 21 years were asked to undertake a standardized test every two years. The test involved an oral glucose tolerance test and some other bodily measurements. Diabetes was diagnosed as per World Health Organization (WHO) Criteria [13]. That defined, if the 2-hour post-load plasma glucose was at least 200 mg/dl (11.1 mmol/l) at any survey test or if the Indian Health Service Hospital serving the community found a glucose concentration of at least 200 mg/dl during the time period of routine medical care. The database is being used for study by the researchers since its formation. Moreover, this data set provides a well-validated data resource to delve into the prediction of the date of onset of diabetes with various techniques.

Set of Features

There are eight features with samples 768 in total. The binary target values 0 and 1 represented 'tested negative' and 'tested positive'

i. *Number of times of pregnancy*
ii. *Plasma glucose concentration at 2hours in an oral glucose tolerance test (GTT)*
iii. *Diastolic blood pressure (mmHg)*
iv. *Triceps skin fold thickness (mm)*
v. *2-hour serum insulin (mu U/ml)*
vi. *Body mass index, i.e. weight/height^2 (Kg/ m^2)*
vii. *Diabetes pedigree function*
viii. *Age (years)*
ix. *Class variable (0 or 1)*

The data set contained all the features including a class variable to defining the class of each sample. There are 500 positive cases and the rest 268 are negative.

3.3 Bayesian Classifier

Using traditional Bayes theorem, Bayesian classifier classifies unknown samples based on maximum likelihood. This is a parametric computational model for solving classification problems. The Bayes theorem states the relationship of events as in (1).

$$P(event\ E|event\ G) = \frac{P(event\ E|event\ G)P(event\ G)}{P(event\ E)} \quad (1)$$

- $P(E)$ and $P(G)$ are the probabilities of event *E* and *G* without regard to each other.
- $P(E \mid G)$, a conditional probability, is the probability of event *E* given that event *G* is true.
- $P(G \mid E)$, a conditional probability, is the probability of event *G* given that event *E* is true.

In the case of naïve Bayesian classifier, the attributes are assumed independent. Not only it is fast and easy to compute but also, it is not sensitive to irrelevant data. From Bayes theorem, it can be deduced that as in (2).

$$P(G|E) \approx P(E|G)P(G) \quad (2)$$

Naïve Bayesian classifier is reliable for classification and has been used for many years. Since the classifier assumes the attributes independent, they do not affect the probability of each other. On the other hand, there are some adversities like, bias, variance and training data noise. Training data noise is sometimes the result of feature extraction and can be reduced by selecting distinguishing features.

The formal definition of Naïve Bayesian classifier concisely stands as in (3):

$$C = arg \max_{k\in\{1,2,\ldots,K\}} P(C_k) \prod_{i=1}^{n} P(x_i|C_k) \quad (3)$$

where,

- $P(C_k)$ = probability of class k (prior probabilty)

- $P(x_i|C_k)$ = probability of feature x_i given class k

(class-conditional probability)

Now, since the attributes are continuous in this case, the likelihood of each class was calculated using the probability density estimations of the attributes. Assuming the distribution as normal, the density estimation function was defined as in (4).

$$\varphi_{\mu,\sigma}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (4)$$

Density function expresses the relative probability of a point where μ is the mean, σ is the standard deviation, $\varphi_{\mu,\sigma}(x)$ is used for calculating $P(x|C_k)$.

The model training was accomplished by estimating the prior $P(C_k)$ and for every attribute A_i, for every attribute value v of A_i estimating $P(A_i = v|C_k)$.

Applying the probabilistic model for a given sample with $(v_1, v_2, v_3, \ldots, v_n)$, the class picked that maximized the value of (5).

$$P(C_k)\prod_{i=1}^{n} P(A_i = v_i|C_k) \quad (5)$$

Algorithm 1 describes our Bayesian classification algorithm.

Algorithm 1: Bayesian classification

i. Input training dataset where each feature vector X_i consists of the feature set $\{x_{i1}, x_{i2}, \ldots, x_{i8}\}$.

ii. Calculate the prior probability $P(C_k)$ for each class k, where k = 0, 1.

iii. Calculate the class-conditional probability $P(X|C_k)$ for each class k, where k = 0, 1.

iv. Input validation dataset.

v. Calculate the generalization error using the validation dataset.

vi. Calculate the posterior probability for each class k considering feature vectors X_i consisting feature set $\{x_{i1}, x_{i2}, \ldots, x_{i8}\}$.

vii. Predict the class of each test sample based on the posterior probability of each class k, greater probability of positive refers the risk of diabetes mellitus whereas that of negative detects the opposite.

4 Implementation

Implementation of the Naïve Bayes classifier is done using the programming language Java running on NetBeans IDE version 8.0.2. The computer system deployed is of the configuration: the processor is Intel core i3 CPU 1.90 GHz, size of installed memory (RAM) is 4GB and operating system (OS) is Windows 8.1 (64-bit OS).

Since it only requires two parameters from the training sample namely mean and standard deviation, we formulated a three-dimensional table to contain the values. The table is referred as “Analysis table” which

contains classes along the rows, features along the columns and levels are preserved for the mean and standard deviation of the specific feature in that specific class. It is to be mentioned here that the last column only contained the prior probability of that class.

The sample data set was inserted into the classifier and trained with different ratios of training and testing samples. As mentioned earlier, the sample data set consists of 768 samples with feature values in the form of numerical values. The sample dataset was divided into training data set and test data set. The means and standard deviations were calculated using standard formulae and stored in the "Analysis table". Once the classifier is initialized and the mean, standard deviations of features are stored, the classifier is ready to classify.

After building the classifier, test samples were input into the classifier one at a time for classifying. We used validation sets of different ratios for validating the training dataset. For classification, the posterior probability of each class for every test sample is calculated. The test sample was assigned to the class with maximum posterior probability.

5 Description of Results Found

We have applied our method to the data set several times. The result we achieved is promising. For calculating accuracy, we followed the simple technique of correctness ratio.

$$accuracy = \frac{no.of\ samples\ correctly\ classified}{no.of samples\ tested} \times 100\%$$

We have tested the samples with different test-train ratios and found a positive result in increasing the training samples. Figure 4 demonstrates the accuracy curve rising along with the increase of training samples. When the test-train ratio was 30%-70%, the result we achieved barely met the expectation. Increasing the training sample percentage to 80%, the accuracy increased a bit. Nevertheless, increasing the training sample to 90% the accuracy had a gradual pickup. It quickly ascends above 85% that is clearly more than previous works in this field. Table I summarizes all the results found. The best result we obtained is 87.28% for the case of 10-fold cross validation.

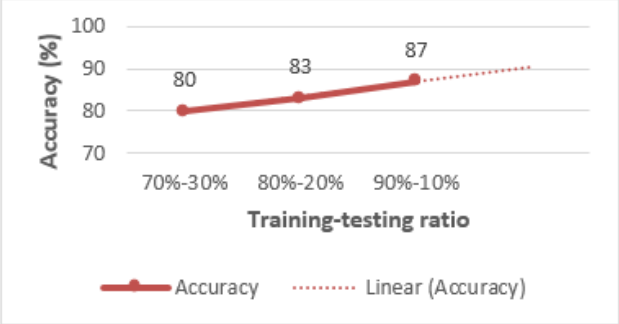


Figure 4. Performance curve

Table 1. Disease Classification Performance

Test-Train Ratio	Accuracy	Average Accuracy
30% - 70%	83.12%	80.74%
	80.95%	
	80.09%	
	78.79%	
20% - 80%	86.36%	82.78%
	82.47%	
	77.92%	
	84.36%	
10% - 90%	88.31%	87.28%
	87.01%	
	86.12%	
	87.69%	

6 Comparative Discussion of Results

Due to the limitation of data, i.e. deficiency of samples, disease prediction does not seem to be much robust. Still, we achieved a promising result by following our approach. In comparison to other works, it would not be an overstatement that our work gives satisfactory results. Yang Guo et al. proposed Naïve Bayes network yielded an accuracy of 72.3% [11]. Jack W. Smith et al. achieved an accuracy of 76% [10]. Again, K. Polat et al. considerably tried to gain more accuracy through SVM. They could achieve an accuracy of 82.05% [7]. S. Karatsiolis achieved 82.2% accuracy using modified support vector machine [14]. Mohammad Amine Chikh et al. raised the mark to 82.69% in their work [15]. H. Kahramanli achieved an accuracy of 84.24% [8] using ANN and FNN. Table II compares different results obtained over the dataset using various methods.

Table 2. Comparative Analysis

Reference	Classifier	Accuracy
[11]	Bayes Belief Network	72.3%
[10]	Neural Network	76%
[7]	Support Vector Machine	82.05%
[14]	Modified Support Vector Machine	82.2%
[15]	Fuzzy K-Nearest Neighbor	82.69%
[8]	ANN and FNN	84.24%
This paper	Proposed Bayesian Classifier	87.28%

7 Conclusion and Future Works

In this paper, we have demonstrated a comprehensive approach to predict the onset of diabetes mellitus based on a well-known data set. Obtained results are very intriguing. Although our findings show a good promise compared to previous works reported in the literature, we still believe there is a chance to improve it further with new techniques like ensemble learning, balancing classes, etc. considering.

As future work, we plan to work with an expert system capable of diagnosing diabetes mellitus robustly. For example, the expansion of the sample data set and reduction of the number of features are required for the improvement of classification. Work is in progress to integrate feature ranking to reduce the feature set and optimize the prediction based on an expanded data set.

REFERENCES

[1] Peter J.F. Lucas & Linda C. van der Gaag(1991)"The Principles of Expert Systems", Amsterdam, Addison-WesleyPublishing Company.

[2] Christopher D. Saudek et al.(2013) "A New Look at Screening and Diagnosing Diabetes Mellitus", *The Journal of Clinical Endocrinology & Metabolism*, vol 93, issue 7.

[3] http://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes

[4] Aibinu, A.M, Salami, MJE, & Shafie, A.A. (2011). "A novel signal diagnosis technique using pseudo complex-valued autoregressive technique". *Expert Systems with Applications, 38*(8), 9063-9069.

[5] Isa NAM, & Mamat WMFW. (2011). "Clustered-Hybrid Multilayer Perceptron network for pattern recognition application". *Applied Soft Computing*, 11(1), 1457-1466.

[6] Polat,K.,& Gunes, S. (2007). "An expert system approach based on principal component analysis and adaptive neuro-fuzzy inference system to diagnosis of diabetes disease". *Digital Signal Processing*, 17(4), 702-710.

[7] Polat, K., Gunes, S.,& Arslan, A.(2008). "A cascade learning system for classification of diabetes disease: Generalized discriminant analysis and least square support vector machine". *Expert Systems with Applications*, 34(1), 482-487.

[8] Kahramanli, H.,& Allahverdi, N. (2008). "Design of a hybrid system for the diabetes and heart diseases". *Expert Systems with Applications*, 35(1-2), 82-89.

[9] Jack W. Smith, JE Everhart, WC Dicksont, WC Knowler, RS Johannes, (1988), "Using the ADAP Learning Algorithm to Forecastthe Onset of Diabetes Mellitus".

[10] Murali S. Shankar (1996), "Using Neural Network to Predict the Onset of Diabetes Mellitus", *Journal of Chemical information and Computer Science,* vol. 36.

[11] Yang Guo, Guohua Bai, Yan Hu (2012), "Using Bayes Network for Prediction of Type-2 Diabetes", *7th International Conference for Internet Technology and Secured Transactions (ICITST)*, London.

[12] Knowler, W.C., DJ. Pettitt, PJ. Savage, and P.H. Bennett 1981. "Diabetes incidence in Pima Indians: contnbutions of obesity and parental diabetes". *Am J Epidemiol* 113:144-156.

[13] World Health Organization, "Report of a Study Group: Diabetes Mellitus. World Health Organization Technical Report Series". Geneva, 727, 1985.

[14] S. Karatsiolis, C. N. Schizas, (2012), "Region based Support Vector Machine algorithm for medical diagnosis on Pima Indian Diabetes dataset",*IEEE 12th International Conference on Bioinformatics & Bioengineering (BIBE)*, pages: 139-144.

[15] Mohamed Amine Chikh, Meryem Saidi, Nesma Settouti (2012), "Diagnosis of Diabetes Diseases Using an Artificial Immune Recognition System2 (AIRS2) with Fuzzy K-nearest Neighbor", *Journal of Medical Systems,* vol 36, issue 5.

SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
ISSN 2054-7390

Robot-Server Architecture for Optimizing Solar Panel Power Output

Ernesto Zamora Ramos, Maria Ramos, Konstantinos Moutafis and Evangelos A. Yfantis
University of Nevada, Las Vegas, Nevada, 89154, USA
zamorara@unlv.nevada.edu, ramosm27@unlv.nevada.edu, moutaf@gmail.com, yfantis@cs.unlv.edu

ABSTRACT

Solar panel facilities for generating electricity have increased exponentially in the recent years. Dust and bird droppings on the solar panels inhibit the energy production. Having people to inspect them and, if needed, clean them is expensive and increases the energy cost. In this research paper we introduce a robot-server architecture for the purpose of inspecting the panels and cleaning them if there is a need for it. The general architecture of the robot consists of a mechanical part, an electromechanical part, an electronic part, and a software part. The mechanical and electromechanical parts consist of an all-terrain vehicle, two electric brushless motors, a telescopic vision system, and telescopic cleaning system with a brush, stepper motors controlling the telescopic vision system, and the telescopic vacuum system with a brushless electric motor. The electronic system consists of three electronic speed controllers, navigation sensors, a computer board, a hard disk, a transceiver, and an antenna for wireless communication. The software consists of a scalable operating system, an intelligent vision system with pattern recognition, a communication software system, an intelligent navigation system, and a file server with a database, TLS security, network communication software based on UDP, and internet communication based on websockets and TCP-IP. In addition to that for street solar lights we designed a PCB board with a sensor that activates a mechanism similar to windshield wipers that cleans the glass of the solar panels powering the lights automatically when needed.

Keywords: Robotic Vision, Solar Power Optimization, Pattern Analysis, Autonomous Vehicles.

1 Introduction

The energy produced by solar panels keeps increasing yearly. Silicon solar panels produce by Panasonic have 22.8% efficiency, making solar panels an economically viable alternative to traditional power. Companies like First Solar it has converted 22.1% of the sunlight energy into electricity using experimental cells made from cadmium telluride. New semiconductor technologies based on Gallium Arsenide and Indium Gallium Nitride (inGan) promise a major improvement over silicon solar panels. Also multi-junction solar panels have higher production of electric energy. Gallium nitride Solar panels are relatively inexpensive, they last for a considerable amount of time, and every year we see new large scale installations as well as smaller for houses and commercial buildings. Many of these large scale installations are in desert environments, where strong winds blow sand and dust onto the panels inhibiting the energy productions. In addition to that birds migrating from colder to warmer climates choose the solar panel sites as a resting place and the bird droppings on the panels inhibit the energy production. Also small

DOI: 10.14738/tmlai.44.2141
Publication Date: 18th August, 2016
URL: http://dx.doi.org/10.14738/tmlai.44.2141

animals with sharp teeth roaming on the solar panel site at night, or using the space under the panels to protect themselves from the hot summer days and the cold winter nights cut the cables with their sharp teeth thus disabling the panels. Finally vandals throwing stones and other objects through the fence could damage the glass or other parts of the panels.

Having a maintenance crew to look after the panels is an expensive proposition which introduces human problems.

Here we introduce a robot that we named "Helios." Its purpose is to inspect every panel, as well as its cable connections, and decide if the panel needs cleaning or not.

All the pertinent information, which includes the panel id, and the details related to the panel status are transmitted wireless to a file server and stored in the data base. For each panel needed cleaning the robot returns during the night when the panel does not produce any energy and cleans the panel using the brush and the vacuum.

The robot (figure 1), consists of a mechanical part, an electromechanical part, an electronics part, and a software part.

The mechanical and electromechanical parts consist of an all-terrain vehicle, two electric brushless motors, a telescopic vision system, and telescopic cleaning system with a brush, stepper motors controlling the telescopic vision system, and the telescopic vacuum system, and a small vacuum system with a brushless electric motor. The vacuum system traces the solar panel from top to bottom and cleans it.

Figure 1: Prototype of the robot Helios with the folding telescopic support of the vision system consisting of four cameras and a noninvasive laser.

The electronics system consists of three electronic speed controllers, a number of ultrasound sensors, navigation sensors, a computer board, a hard disk, a transceiver, and an antenna for wireless communications.

The software consists of a scalable operating system, an intelligent vision system with pattern recognition, a communication software system, and an intelligent navigation system.

The computer mother board has a solid state high capacity hard disc to store the operating system, with several gigabytes of memory, a light SQL database with the location of each panel and other panel

information. The wireless communication system includes a transceiver, an antenna, an MCU, and memory, and a network software system for communication with the file server. The file server contains the database holding information about each panel, and about each robot.

In this research paper we describe the general architecture of the robot, which includes the mechanical design, electronics and software architecture. This research paper is structured as follows: The abstract is followed by the introduction which is followed by the background information, which is followed by the Robot-server architecture, ending with the conclusion, and the references.

2 Background information

Renewable energy represents a large number of energy solutions promising to replace or reduce the dependency of energy sources of the past that are not friendly to the environment, and are limited so in the future we run the risk to run out of these resources. As more emphasis is placed to research and development of new renewable energy technologies that are more efficient, cost effective, and non-environmentally toxic to provide energy for our energy needs in our houses, business, street lights, public buildings, transportation, communications, and help create clean smart cities with cleaner air, that provide a healthier environment.

Photogrammetry (photo-light, gram-drawing, metry-measuring) has a Greek derivation, and is the practice of determining the geometric properties of objects from photographic images. It is dated back to nineteenth century when film photography started. The process is as simple as getting the distance between two points on a plane parallel to the photographic image plane. Work in stereo photogrammetric image enhancement, image processing, and stereo vision started in the later part of the last century.

Our work pertains to robot-server communication, specialized robot architectures, robot vision, machine intelligence and pattern recognition [1-8], robot navigation, and autonomous machines.

The vision system used in our robot system consists of four cameras on a cross framework. The horizontal and vertical dimensions of the cross at the default state are equal. The cross has telescopic components so that the cameras could be at different distances from the center. In the center of the cross there is a non-invasive laser. The cross is supported by a mechanism that allows the system of four cameras to pan and tilt. Each camera is supported by its own pan and tilt mechanism. The Mathematics and analytics of the computer vision system are presented in this research as well as the architecture of the robot.

3 The Robot-Server Architecture

The Robot-Server architecture is very similar to the server-client paradigm. The server has a data base that includes each robot, each panel, and the pathway to each panel, from an origin. The server also includes the local area network of the server and robots, the communication software between the server and each robot. The robots include the image processing, classification, and the software driving the robot, and enabling the robot to perform cleaning, and other functions. The majority of the effort is in the software, making the robot to be a specialized computer on wheels. The hardware consists of three parts.

3.1 The Mechanical Component

The mechanical component comprises the vehicle which is an all-terrain using an army-tank-like continuous track, with two brushless motors, one in each front wheel. Each one of the motors is controlled by a separate electronic speed controller. The main reason for this is to enable the robot to make turns. Thus in order for the robot to turn left; we increase the speed on the right motor and decrease the speed

on the left. The mechanical part also includes a telescopic vision system which provides the input to the intelligent software that understands a panel's boundaries and decides if a panel is clean or needs to be cleaned. The mechanical part also includes the vacuum and a brush used to loosen material on the panel in order for the vacuum to clean the panel.

The electromechanical system includes the two brushless motors of the vehicle part of the robot, the stepper motors of the telescopic vision system, as well as the stepper motors of the telescopic vacuum system, and the brushless motor of the vacuum system.

3.2 The Electronic Component

The electronic part consists of a printed circuit board (PCB) connected to the four cameras via four BNC connections, having a number of sensors used as part of the navigation system, GPS, accelerometer, magnetometer, a DSP that takes as input the images obtained by the four cameras via the BNC connections, stores the images in four memory chips on board, performs the classification algorithm and makes a decision if the panel is clean or needs cleaning. The decision is passed to the transceiver on board to transmit it wireless to the file server. The PCB board also contains a control system that uses an ARM chip to compress the images obtained by the four cameras, passes them to the transceiver on board which transmits them to the file server; a transceiver, a voltage amplifier that amplifies the voltage from 1.5V to 12V, and an antenna. The PCB board is connected to a computer board via a PCI express connection.

3.3 The Vision System

The software consists of the classification algorithm, that takes as input the images of the panel obtained by the four cameras, applies the classification algorithm and decides if the panel needs cleaning. The vision system also inspects the electric cables underneath the panel and decides if all connections are good or not.

The vision system is part of the robot and it consists of four identical cameras and a non-invasive laser. Figure 2, depicts the schematics of our system. The cameras are mounted on a frame having a cross configuration. Each leg of the cross is telescopic having the ability to increase or decrease the distance of the camera from the laser so that will decrease or eliminate occlusions. The distance of each camera from the center of the cross is controlled by a stepper motor and it is always known. The noninvasive laser is in the middle of the cross and is equidistant to the four cameras. When distances are to be resolved the noninvasive laser is activated and its light is registered by each one of the four cameras. The cameras are parallel to one another and also parallel to the laser. During calibration for every pixel in the image space registering the laser light dot on the object, the angle between the line defined by the image center and the pixel, and the line defined by the pixel and the laser dot on the object, is computed and stored in a lookup table. Thus during the focus on a panel if for a camera the laser dot is registered by a certain pixel then we know the angle formed by the line between the pixel and the laser dot and the line between the pixel and the camera center.

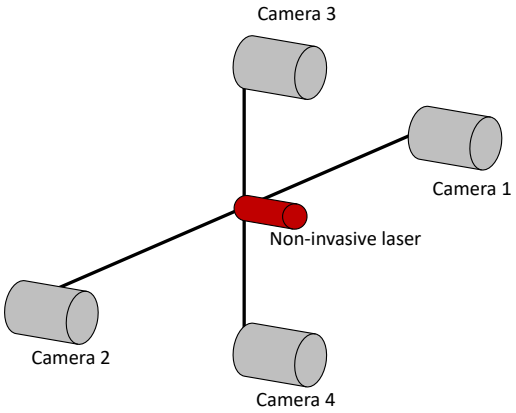


Figure 2: The schematics of our vision system. The vision system consists of four identical cameras and a noninvasive laser. The purpose of the system is to resolve distances, and enable the creation of 3-D vision.

Figure 3 shows each of the four cameras with each own local coordinate system. In the default state the laser and the cameras are parallel and the distance of each camera from the laser is fixed. The default state is the one we use in order to position the camera system at a fixed distance from the panel. In this state the laser dot has exactly the same *Z* coordinate for each one of the four local camera coordinate systems.

The *Z* coordinate for a pinhole camera (figure 4), is given by equation 1. If *B* is the known distance between the focal points of cameras 1 and 2 then equation 4 gives an estimate of the distance of the laser focal point to the laser dot on the panel. In a similar way we can obtain another estimate of *Z* from the cameras 3 and 4. Two estimates of *Z* can be obtained from cameras 1 and 3, another two from cameras 2 and 3, another two estimates of *Z* from cameras 1 and 4, and finally another two estimates of *Z* from cameras 2 and 4. An estimate of *Z* can be obtained from camera 1 and the laser, similarly another from camera 2 and the laser, from camera 3 and the laser, and from camera 4 and the laser.

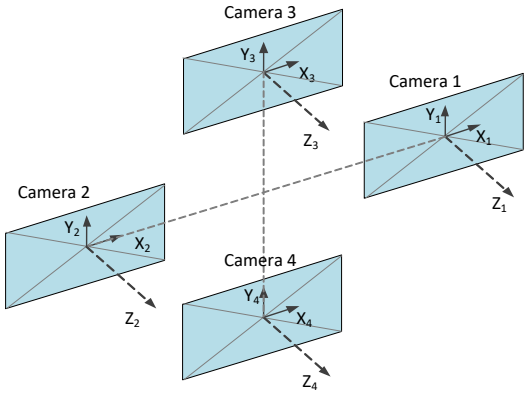


Figure 3: Each one of the cameras has a local coordinate system (X_i, Y_i, Z_i), and an image space coordinate system (x_i, y_i), $i \in \{1,2,3,4\}$. The local coordinates can easily be transformed to a global coordinate system via translation and rotation.

Thus, 14 estimates of the distance of the laser focal point from the laser dot on the panel can be obtained. All these estimates are slightly different due to the noise in the system. The average $\overline{Z}$ of these fourteen estimates is a more accurate estimate of the distance of the vision system from the panel. The DSP of the

PCB board computes this distance relatively fast and positions the vision system at a fixed distance above the panel. This distance is the same at every inspection of every panel. The details of the geometry and formulas are given below.

The pinhole camera model can also represent the modern CCD or CMOS cameras with the chip replacing the film and the center of the lens replacing the pinhole. In figure 4, *O* is the center pixel of the imager chip, and also the center of the local coordinate system. $L(0,0,f)$ is the lens center, $P(X,Y,Z)$ is a point in the space projected to the point $\hat{P}(x,y)$ on the imager. Then from the similar triangles ΔALB and ΔOaL, if $\lambda = \overline{LO}$, $\overline{Oa} = x$, $\overline{LB} = Z - \lambda$, we have:

$$-\frac{Z-\lambda}{\lambda} = \frac{X}{x}, \text{ or } Z = \lambda\left(1 - \frac{X}{x}\right) \tag{1}$$

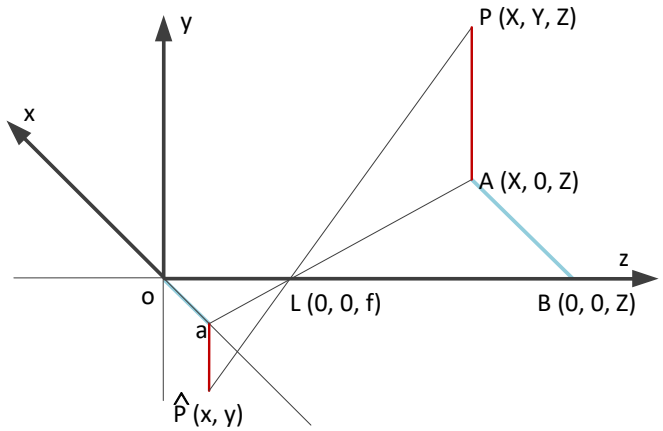


Figure 4: Pinhole camera model. L is the camera pinhole or the center of the lens. *O* is the center of the imager, as well as the origin of the local coordinate system (X,Y,Z), and the image coordinate system (x,y). The point $P(X,Y,Z)$ is projected to the point $\hat{P}(x,y)$.

From equation 1 we have:

$$Z = \lambda\left(1 - \frac{X_1}{x_1}\right) \tag{2}$$

$$Z = \lambda\left(1 - \frac{X_2}{x_2}\right) = \lambda\left(1 - \frac{X_1 + B}{x_2}\right) \tag{3}$$

Thus from 2 and 3 we get:

$$Z = \lambda\left(1 - \frac{B}{x_2 - x_1}\right) \tag{4}$$

Due to the noise in the system we obtain four different estimates of *Z* from the four laser-camera geometries: two estimates using the geometry of two horizontal and two vertical cameras. Eight different estimates of *Z* can be also obtained using the geometry of any two adjacent cameras. Each one of these estimates is a random variable with mean Z_t, the true value of the distance, and variance σ_i^2, $i \in \{1,2,...,14\}$. According to the central limit theorem the average $\bar{Z}$ of these estimates of the distance is normally distributed with mean Z_t, where Z_t is the true distance, and variance

$$\sigma_{\bar{Z}}^2 = \frac{\sum_{i=1}^{14} \sigma_i^2}{14}.$$

Let S_Z be an estimate of the standard deviation of the random variable *Z* based on the estimates of the true distance, and if we denote by Z_t the true value of *Z* then the statistic:

$$\frac{\bar{Z} - Z_t}{\frac{S_Z}{\sqrt{14}}}$$

has the *t* distribution with 13 degrees of freedom. Therefore

$$\bar{Z} - t_{1-\frac{\alpha}{2},13} < Z_t < \bar{Z} + t_{1-\frac{\alpha}{2},13} \tag{5}$$

with probability $1-\alpha$

Formula 5 provides a measure of how accurate the distances estimated by our system are. For example if $\bar{Z} = 100\text{cm}$ and $S_Z = 3\text{cm}$, then with probability 0.95 (95%) the true distance Z_t, is $98.27\text{cm} < Z_t < 101.73\text{cm}$.

4 Experimental Results

As the robot prototype is still in development, we have directed experimentation and evaluation to testing completed components of the software and hardware.

Table 1 Results of equation 6 from applying Jackknife test on all three groups of sample data [1].

Group	TN	FN	TP	FP	Accuracy (%)	Misclassification Error
1	12	0	12	0	100	0
2	17	3	19	1	90	0.10
3	17	2	19	0	94.4	0.0526

The energy produced by solar panels declines in proportion to the amount of light blocked by the deposits of dust and other contaminants accumulating on the surface of the photovoltaic cells.

Our current work regarding the classification system employed by the robot to determine if a panel is clean or not is based on the Mahalanobis distance, which is the relative, statistical measure of the data

point's distance from a common point. The classifier we developed for the system recognizes the state of the panel with accuracy above 90%.

To evaluate the accuracy of the classification system we obtained sample images from solar panels using a camera configuration similar to what will be part of the finalized vision system of the robot. We decided to employ three groups of training data, where each group contains one clean sample set and one dirty sample set.

1) The first group contains data from the same panel.
2) The second group builds upon the first group by incorporating data from another panel with similar photovoltaic cell structure.
3) The third group does not build upon the first and second group. Instead, it incorporates data from two panels of similar characteristics, but with a lighter shade of blue than the panels from the other groups.

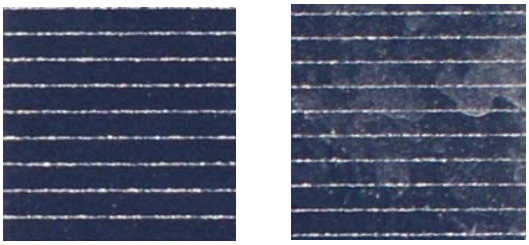


Figure 5: Solar panel samples: left) clean panel; right) dirty panel [1].

See figure 5 for two samples taken by the vision system of clean and dirty solar panels.

Then we utilized the jackknifing technique to estimate the precision of the classifier through the formula:

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP} \tag{6}$$

where TN (true negative)/FN (false negative) are the number of samples correctly/incorrectly classified as clean, and TP (true positive)/FP (false positive) are the number of samples correctly/incorrectly classified as dirty [1].

Table 1 shows the experimental results obtained from the sample data.

Further development, experimentation and evaluation of other components are still in progress for this project. The results of the vision system are, however, very promising.

5 Conclusion

In this research paper we describe the architecture of a Robot-file server system still in experimental and development form, used to recognize if solar panels are dirty and lose energy above a critical value. In such cases, the robot cleans the affected panels and transmits this information to the file server in order to update the flag on record related to the panel in the database to "cleaned" along with the date it was cleaned.

The robot software includes a scalable operating system, along with an intelligent navigation system, classification software, software driving the cleaning mechanism, and communication software that include web socket communication, and network socket communication.

ACKNOWLEDGEMENT

This Research was supported by the National Science Foundation (NSF) Nexus of Food, Energy and Water grant No. IIA-1301726.

REFERENCES

[1]. E. Zamora Ramos, S. Ho, and E. A. Yfantis, *Using spectral decomposition to detect dirty solar panels and minimize the impact on energy production.* Journal of Advances in Image and Video Processing, 2015. 3(6): p. 1-12.

[2]. E. A. Yfantis, *Telemedicine: The present and its future.* In *Plenary Lecture: 14th International Conference in Applied Computer Science.*

[3]. E. A. Yfantis, *Dynamic redundancy bit allocation and packet size to increase throughput in noisy real time video wireless transmission.* The Journal of Combinatorial Mathematics and Combinatorial Computing JCMCC 86, 2013. p. 163-169.

[4]. E. A. Yfantis et al, *Pollution detection in urban areas using the existing camera networks.* International Journal of Multimedia Technology, 2013. 3(3): p. 98-102.

[5]. E. A. Yfantis and A. Fayed, *Authentication and secure robot communication.* International Journal of Advanced Robotic Systems, 2014. 11: p. 1-6.

[6]. E. A. Yfantis and A. Fayed, *A camera system for detecting dust and other deposits on solar panels.* Journal of Advances in Image and Video Processing, 2014. 2(5): p. 1-10.

[7]. E. A. Yfantis and A. Fayed, *Robot vision and distance estimation.* In *14th International Conference in Applied Computer Science*, Cambridge, MA, Jan 2014. p. 215-220.

[8]. E. Zamora Ramos et al. *A Robot Architecture for Detecting Dust and Cleaning Solar Panels*. In *31st International Conference on Computers and Their Applications, CATA 16*, Las V

SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
ISSN 2054-7390

Support Vector Machine Regression and Artificial Neural Network for Channel Estimation of LTE Downlink in High-Mobility Environments

[1*]Anis Charrada and [2]Abdelaziz Samet
[1]SERCOM Laboratory, Tunisia Polytechnic School, University of Carthage, Tunis, Tunisia;
[*]Tunisian Military Academy, Tunisia;
[2]INRS, EMT Center, 800 de la Gauchetière W., Suite 6900, Montreal, QC, H5A 1K6, CANADA;
anis.charrada@gmail.com

ABSTRACT

In this paper we apply and assess the performance of support vector machine regression (SVR) and artificial neural network (ANN) channel estimation algorithms to the reference signal structure standardized for LTE Downlink system. SVR and ANN where applied to estimate real channel environment such as vehicular A channel defined by the International Telecommunications Union (ITU) in the presence of nonlinear impulsive noise. The proposed algorithms use the information provided by the received reference symbols to estimate the total frequency response of the time variant multipath fading channel in two phases. In the first phase, each method learns to adapt to the channel variations, and in the second phase it predicts all the channel frequency responses. Finally, in order to evaluate the capabilities of the designed channel estimators, we provide performance of SVR and ANN, which is compared with traditional Least Squares (LS) and Decision Feedback (DF). The simulation results show that SVR has a better accuracy than other estimation techniques.

Keywords: Complex SVR; ANN; nonlinear impulsive noise; OFDM and LTE.

1 Introduction

The Long Term Evolution (LTE) is a step towards the fourth generation (4G) of mobile radio technologies to obtain higher throughput and to increase the spectral efficiency. Fourth-generation broadband wireless multiple access systems have data rate specifications on the order of hundreds of Mb/s. For an LTE system with 20 MHz bandwidth, the objective is for the Downlink (DL) and Uplink (UL) peak data rates to require 100 and 50 Mb/s, respectively [1].

LTE Downlink uses Orthogonal Frequency Division Multiplex Access (OFDMA) radio interface, that is support high data rate capabilities and it is more resilient against severe channel conditions. OFDMA technique essentially distributes the symbols of a large number of carriers. By implementing this new access technique in the context of mobile broadband transmission, new approaches for time and frequency equalization, synchronization and channel estimation are needed.

Because channel estimation is an important concern of LTE DL, some research results have been published, including Least Squares (LS) and Minimum Mean Square Error (MMSE)-based techniques such

DOI: 10.14738/tmlai.44.2145
Publication Date: 25th August, 2016
URL: http://dx.doi.org/10.14738/tmlai.44.2145

as [2] and [3] where the authors have studied the performance of two linear channel estimators, the Least Squares Error (LSE) and the Linear MMSE (LMMSE).

In LTE with a time variant highly selective multipath fading channel, where complicated nonlinearities can be found (the channel variations in time and in frequency domain are nonlinear in addition to the presence of impulsive noise), the accuracy of estimation can significantly decrease by applying the linear process.

ANN can perform complex mapping between its input and output space and are capable of forming complex decision regions with nonlinear decision boundaries [4]. Further, these networks of different architectures have found successful application in channel estimation problems because of their nonlinear characteristics. ANN is proposed as a channel estimator for QPSK and QAM constellation, respectively in [5] and [4].

In this paper, we designed first a Back Propagation Algorithm BPA-based ANN channel estimation technique for LTE-OFDM system over frequency selective multipath fading channel in the presence of nonlinear impulsive noise interfering with reference symbols under high mobility conditions.

In addition, we designed a complex Support Vector Machine Regression (SVR) based on Radial Basis Function (RBF) kernel for channel estimation in LTE DL that maps the input vector from a finite-dimensional space (the input space) to a higher dimensional Hilbert space (can be infinity) which it is provided with a dot product. Training SVR approach is used for channel estimation of highly selective multipath channels for OFDMA systems where the LS algorithm was applied in the training step as a channel estimator: it uses the obtained estimations as a dataset for training. The idea is to exploit the information supplied by the pilot symbols in order to estimate the channel frequency response.

The organization of this paper is as follows. In section 2, description of LTE Downlink system model is given. ANN based LTE channel estimator is introduced in section 3. In section 4, a nonlinear channel estimator based on the complex SVR is provided. Simulation results are offered in section 5. Finally, section 6 concludes the paper.

2 LTE Downlink System Model

The LTE DL system is based on the OFDMA air interface transmission scheme. Figure 1 shows the block diagram of the baseband equivalent system model.

Let us consider an LTE Downlink system which comprises N subcarriers, occupying a bandwidth *B*. The corresponding OFDM system consists firstly of mapping binary data streams into complex symbols by means of QAM modulation. Then data are transmitted in frames by means of serial-to-parallel conversion. Some pilot symbols are inserted into each data frame which is modulated to subcarriers through the Inverse Discret Fourier Transform (IDFT). These pilot symbols are inserted for channel estimation purposes. The IDFT is used to transform the data sequence $X(k)$ into time domain signal.

One guard interval (GI) is inserted between every two OFDM symbols in order to eliminate Inter-Symbol Interference (ISI). This guard time includes the cyclically extended part of the OFDM symbol in order to preserve orthogonality and eliminate Inter-Carrier Interference (ICI). It is well known that if the channel impulse response has a maximum of L resolvable paths, then the GI must be at least equal to L [6]. Thus, each OFDM symbol is transmitted in time T and includes a cyclic prefix of duration T_{cp}. Therefore, the duration of each OFDM symbol is $T_u = T - T_{cp}$. Every two adjacent subcarriers are spaced by $\delta f = 1/T_u$. The output signal of the OFDM system is converted into serial signal by parallel to serial converter. A

complex Additive White Gaussian Noise (AWGN) process $N\left(0,\sigma_{w_g}^2\right)$ with power spectral density $N_0/2$ is added through a frequency selective time varying multipath fading channel.

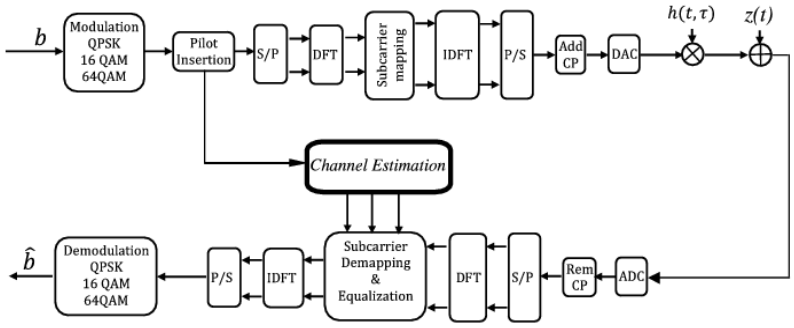


Figure 1: Block diagram of the baseband equivalent system model.

In a practical environment, impulsive noise can be present, and then the channel becomes nonlinear with non Gaussian impulsive noise. The impulsive noise can significantly influence the performance of the OFDM communication system for many reasons. First, the time of the arrival of an impulse is unpredictable and shapes of the impulses are unknown and they vary considerably. Moreover, impulses usually have very high amplitude, and thus high energy, which can be much greater than the energy of the useful signal [7].

The impulsive noise is modeled as a Bernoulli-Gaussian process and it was generated with the Bernoulli-Gaussian process function $i(n) = v(n)\,\lambda(n)$ where $v(n)$ is a random process with Gaussian distribution and power σ_{BG}^2, and $\lambda(n)$ is a random process with probability [8]

$$P_r(\lambda(n)) = \begin{cases} p & \lambda = 1 \\ 1-p, & \lambda = 0. \end{cases} \tag{1}$$

At the receiver side, and after removing guard time, the discrete-time baseband OFDM signal can be expressed as

$$y(n) = \sum_{k\in\{\Omega_P\}} X^P(k)H(k)e^{j\frac{2\pi}{N}kn} + \sum_{k\notin\{\Omega_P\}} X^D(k)H(k)e^{j\frac{2\pi}{N}kn} + w_g(n) + i(n) \tag{2}$$

where Ω_P the subset of N_P pilot subcarriers, $X^P(k)$ and $X^D(k)$ are complex pilot and data symbol respectively, transmitted at the k^{th} frequency and $H(k) = DFT_N\{h(n)\}$ is the channel's frequency response at the k^{th} subcarrier. Note that, pilot insertion in the subcarriers of every OFDM symbol must satisfy the demand of the sampling theory and uniform distribution [9].

Assuming that ISI are eliminated after DFT transformation, therefore $y(n)$ becomes

$$Y(k) = X(k)H(k) + W_G(k) + I(k) = X(k)H(k) + e(k), \quad k = 0,\cdots,N-1 \tag{3}$$

where $e(k)$ is the residual noise which represents the sum of the AWGN noise $W_G(k)$ and impulsive noise $I(k)$ in the frequency domain, respectively.

Equation (3) may be presented in matrix notation as follows:

$$Y = \boldsymbol{X}\boldsymbol{F}h + W + I = \boldsymbol{X}H + e \tag{4}$$

where

$$\boldsymbol{X} = diag\big(X(0), X(1), \cdots, X(N-1)\big)$$

$$Y = [Y(0), \cdots, Y(N-1)]^T$$

$$W_G = [W_G(0), \cdots, W_G(N-1)]^T$$

$$I = [I(0), \cdots, I(N-1)]^T$$

$$H = [H(0), \cdots, H(N-1)]^T$$

$$e = [e(0), \cdots, e(N-1)]^T$$

$$\boldsymbol{F} = \begin{bmatrix} F_N^{00} & \cdots & F_N^{0(N-1)} \\ \vdots & \ddots & \vdots \\ F_N^{(N-1)0} & \cdots & F_N^{(N-1)(N-1)} \end{bmatrix}$$

and $$F_N^{i,k} = \left(\frac{1}{\sqrt{N}}\right) exp^{-j2\pi\left(\frac{ik}{N}\right)}. \quad (5)$$

3 ANN Estimation

Artificial Neural Networks (ANN) are one of the widespread branches of artificial intelligence. They have very simple neuron-like processing elements (called artificial neurons or nodes) connected to each other by weighting. The weights on each connection can be dynamically adjusted until the desired output is generated for a given input. An artificial neuron model consists of a linear combination followed by an activation function. Different types of activation functions can be utilized for the network; nevertheless, the common ones, which are sufficient for most applications, are the sigmoid and hyperbolic tangent functions [10].

In the input and hidden layers, neural networks contain neurons with nonlinear activation functions, whereas in the output layer, neural networks contain neurons with linear activation functions. ANN has multi-layer perceptron (MLP) structure, which uses forward propagation neural network. Various training algorithms exist for MLP. In this work, we used the Scaled Conjugate Gradient Backpropagation (SCG) algorithm which is based on the conjugate directions. Block diagram of the proposed ANN based technique is shown in Figure 2.

The estimator uses the information provided by the reference symbols to estimate the total channel frequency response. At the beginning of the estimation process, the complex signal is split into two parts: real and imaginary. These parts are normalized between -1 and +1 before training.

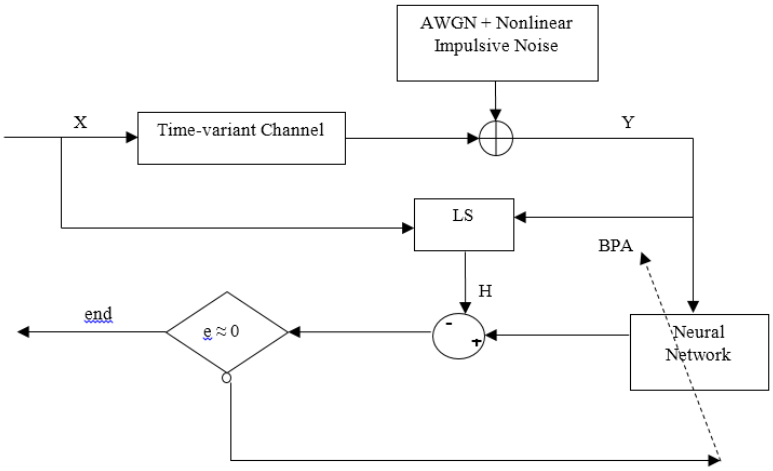


Figure 2: ANN channel estimation trained by Back-Propagation Algorithm.

The adopted architecture of neural network is chosen after multiple tests of convergence by minimizing the learning time and keeping low implementation complexity.

The output of a single neuron is given by the following equation:

$$\hat{A}_j = f\left(\sum_{i=0}^{2N_P-1} w_{j,i}\, P_i + b_j\right) \tag{6}$$

where, $\hat{A}_j$ is the neuron output in the range of $(0 \le j \le 2N-1)$ since there are N real part and N imaginary part of $\hat{A}_j$, $w_{j,i}$ is a value of the synaptic weight connecting the stimulus i to the neuron j, P_i, is the input stimulus, b_j is the bias of neuron j, and f is the neuron output sigmoid function. The weights $w_{j,i}$ are updated with the SCG backpropagation algorithm.

The estimator training operation consists of changing the values of interconnection weights using learning algorithms for obtaining the desired performance. The learning algorithm in our neural network is the efficient gradient back propagation based on the minimization of the average square error (for all $2N$ output neurons) expressed as

$$e = \frac{1}{N_l}\sum_{l=0}^{N_l-1}\sum_{j=0}^{2N-1}\left(e_j^l\right)^2 \tag{7}$$

where e_j^l represents the error on the j^{th} neuron output from the l^{th} example of training set.

After completing the training phase, the network uses the input data from the pilot channels to estimate all the data channels. Subsequently, the equalization is followed by a decision estimate of the OFDMA symbols. For a single training operation, the neural network estimates a large number of OFDM symbols corresponding to several radio frames LTE.

4 Complex SVR estimation

First, let the OFDM frame contains Ns OFDM symbols which every symbol includes N subcarriers. Then, we exploit the index of the pilots in the OFDM symbols in order to estimate the channel frequency

responses at these positions. The transmitting pilot symbols are $\boldsymbol{X}^P = diag(X\ (i, m\,\Delta P)), m = 0,1,\cdots, N_P - 1$, where i and m are labels in time domain and frequency domain respectively, and ΔP is the pilot interval in frequency domain.

The proposed channel estimation method is based on complex SVR algorithm which has two separate phases: learning phase and estimation phase. In training phase, we estimate first the subchannels pilot symbols according to LS criterion to strike $min\ [(Y^P - \boldsymbol{X}^P \boldsymbol{F} h)\ (Y^P - \boldsymbol{X}^P \boldsymbol{F} h)^H]$ [11], as

$$\widehat{H}^P = \boldsymbol{X}^{P\ -1}\ Y^P \tag{8}$$

where $Y^P = Y\ (i, m\,\Delta P)$ and $\widehat{H}^P = \widehat{H}\ (i, m\,\Delta P)$ are the received pilot symbols and the estimated frequency responses for the i^{th} OFDM symbol at pilot positions $m\ \Delta P$, respectively.

Then, in the estimation phase and by the interpolation mechanism, frequency responses of data subchannels will be predicted based on the regression model built in training space. Therefore, frequency responses of all the OFDM subcarriers are

$$\widehat{H}\ (i, q\ ) = f\left(\widehat{H}^P (i, m\,\Delta P)\right) \tag{9}$$

where $q = 0, \cdots, N - 1$, and $f(\cdot)$ is the interpolating function, which is determined by the nonlinear complex SVR approach.

In fact, in a nonlinear deep fading channel, it is necessary to apply the nonlinear complex SVR technique for channel estimation since SVM is superior in solving nonlinear, small samples and high dimensional pattern recognition [12]. The basic idea of Mercer's theorem is that a vector in the input space (finite dimensional space) can be mapped to a higher dimensional feature space $\mathcal{H}$ (possibly infinity) by means of nonlinear transformation $\boldsymbol{\varphi}$. However, this transformation usually remains unknown. Hence, only the dot product of the corresponding space is required and can be stated as a function of the input vectors as following:

$$\boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{\varphi}(\boldsymbol{x}_i), \boldsymbol{\varphi}(\boldsymbol{x}_j) \rangle \tag{10}$$

Such spaces are known as Reproducing Kernel Hilbert Spaces (RKHS) where $\boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the kernel that satisfy the conditions of Mercer's theorem (it is the inner product of a Hilbert space). In this paper, we are using the Radial Basis Function (RBF) which is expressed as

$$\boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}\right) \tag{11}$$

After mapping the input vectors to a higher-dimensional feature space using the nonlinear transformation $\boldsymbol{\varphi}$, the linear regression function can be stated as follows:

$$\widehat{H}(m\,\Delta P) = \boldsymbol{w}^{\boldsymbol{T}} \boldsymbol{\varphi}(m\,\Delta P) + b + e_m, \qquad m = 0, \cdots, N_P - 1 \tag{12}$$

where $\boldsymbol{w}$ is the weight vector, b is the bias term well known in the SVM literature and residuals $\{e_m\}$ account for the effect of both approximation errors and noise. In the SVM framework, the optimality criterion is a regularized and constrained version of the regularized Least Squares criterion.

In general, SVM algorithms minimize a regularized cost function of the residuals, usually the Vapnik's $\varepsilon - insensitivity$ cost function [8].

A robust cost function is introduced to improve the performance of the estimation algorithm which is ε -Huber robust cost function [11] [13], given by

$$\mathcal{L}^{\varepsilon}(e_m) = \begin{cases} 0, & |e_m| \le \varepsilon \\ \frac{1}{2\gamma}(|e_m| - \varepsilon)^2, & \varepsilon \le |e_m| \le e_C \\ C(|e_m| - \varepsilon) - \frac{1}{2}\gamma C^2, & e_C \le |e_m| \end{cases} \tag{13}$$

where $e_C = \varepsilon + \gamma C$, ε is the insensitive parameter which is positive scalar that represents the insensitivity to a low noise level, parameters γ and C control essentially the trade-off between the regularization and the losses, and represent the relevance of the residuals that are in the linear or in the quadratic cost zone, respectively. The cost function is linear for errors above e_C, and quadratic for errors between ε and e_C. Note that, errors lower than ε are ignored in the $\varepsilon - insensitivite$ zone. On the other hand, the quadratic cost zone uses the $L_2 - norm$ of errors, which is appropriate for Gaussian noise, and the linear cost zone limits the effect of sub-Gaussian noise [14]. Therefore, the ε -Huber robust cost function can be adapted to different types of noise.

Let us assume that $\mathcal{L}^{\varepsilon}(e_m) = \mathcal{L}^{\varepsilon}\big(\mathcal{R}(e_m)\big) + \mathcal{L}^{\varepsilon}\big(\Im(e_m)\big)$ since $\{e_m\}$ are complex, where $\mathcal{R}(\cdot)$ and $\Im(\cdot)$ represent real and imaginary parts, respectively. Now, the SVR primal problem can be stated as minimizing

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{1}{2\gamma}\sum_{m\in I_1}(\xi_m + \xi_m^*)^2 + C\sum_{m\in I_2}(\xi_m + \xi_m^*) + \frac{1}{2\gamma}\sum_{m\in I_3}(\zeta_m + \zeta_m^*)^2 + C\sum_{m\in I_4}(\zeta_m + \zeta_m^*) - \frac{1}{2}\sum_{m\in I_2, I_4}\gamma C^2 \tag{14}$$

constrained to

$$\begin{aligned} \mathcal{R}\big(\widehat{H}(m\,\Delta P) - \boldsymbol{w^T}\boldsymbol{\varphi}(m\,\Delta P) - b\big) &\le \varepsilon + \xi_m \\ \Im\big(\widehat{H}(m\,\Delta P) - \boldsymbol{w^T}\boldsymbol{\varphi}(m\,\Delta P) - b\big) &\le \varepsilon + \zeta_m \\ \mathcal{R}(-\widehat{H}(m\,\Delta P) + \boldsymbol{w^T}\boldsymbol{\varphi}(m\,\Delta P) + b) &\le \varepsilon + \xi_m^* \\ \Im(-\widehat{H}(m\,\Delta P) + \boldsymbol{w^T}\boldsymbol{\varphi}(m\,\Delta P) + b) &\le \varepsilon + \zeta_m^* \\ \xi_m^{(*)}, \zeta_m^{(*)} &\ge 0 \end{aligned} \tag{15}$$

for $m = 0, \cdots, N_P - 1$, where ξ_m and ξ_m^* are slack variables which stand for positive and negative errors in the real part, respectively. ζ_m and ζ_m^* are the errors for the imaginary parts. I_1, I_2, I_3 and I_4 are the set of samples for which:

I_1 : real part of the residuals are in the quadratic zone;

I_2 : real part of the residuals are in the linear zone;

I_3 : imaginary part of the residuals are in the quadratic zone;

I_4 : imaginary part of the residuals are in the linear zone.

To transform the minimization of the primal functional (14) subject to constraints in (15), into the optimization of the dual functional, we must first introduce the constraints into the primal functional by means of Lagrange multipliers to obtain the primal-dual functional. Then, by making zero the primal-dual functional gradient with respect to ϖ_i, we obtain an optimal solution for the weights

$$\boldsymbol{w} = \sum_{m=0}^{N_P-1} \psi_m \, \boldsymbol{\varphi}(m\,\Delta P) = \sum_{m=0}^{N_P-1} \psi_m \, \boldsymbol{\varphi}(P_m) \tag{16}$$

where $\psi_m = (\alpha_{\mathcal{R},m} - \alpha^*_{\mathcal{R},m}) + j(\alpha_{I,m} - \alpha^*_{I,m})$ with $\alpha_{\mathcal{R},m}, \alpha^*_{\mathcal{R},m}, \alpha_{I,m}, \alpha^*_{I,m}$ are the Lagrange multipliers (or dual variables) for real and imaginary part of the residuals and $P_m = (m\,\Delta P)$, $m = 0, \cdots, N_P - 1$ are the pilot positions.

In order to solve the dual function, we define the Gram matrix as

$$\boldsymbol{G}(u,v) =< \boldsymbol{\varphi}(P_u), \boldsymbol{\varphi}(P_v) >= K(P_u, P_v) \tag{17}$$

where $K(P_u, P_v)$ is a Mercer's kernel which represents the RBF kernel matrix which allows obviating the explicit knowledge of the nonlinear mapping $\boldsymbol{\varphi}(\cdot)$. A simplified compact form of the functional problem can be stated in matrix format by placing optimal solution $\boldsymbol{w}$ into the primal dual functional and grouping terms. Subsequently, the dual problem consists of maximizing

$$max \; -\frac{1}{2}\boldsymbol{\psi}^H(\boldsymbol{G} + \gamma \boldsymbol{I})\boldsymbol{\psi} + \mathcal{R}\left(\boldsymbol{\psi}^{\boldsymbol{H}} Y^P\right) - (\boldsymbol{\alpha}_{\boldsymbol{\mathcal{R}}} + \boldsymbol{\alpha}^*_{\boldsymbol{\mathcal{R}}} + \boldsymbol{\alpha}_{\boldsymbol{I}} + \boldsymbol{\alpha}^*_{\boldsymbol{I}})\mathbf{1}\mathcal{E} \tag{18}$$

constrained to

$$0 \leq \alpha_{\mathcal{R},m}, \alpha^*_{\mathcal{R},m}, \alpha_{I,m}\,, \alpha^*_{I,m} \leq C \tag{19}$$

where $\boldsymbol{\psi} = [\psi_0, \cdots, \psi_{N_P-1}]^T$; **I** and **1** are the identity matrix and the all-ones column vector, respectively; $\boldsymbol{\alpha}_{\boldsymbol{\mathcal{R}}}$ is the vector which contains the corresponding dual variables, with the other subsets being similarly represented. The weight vector can be obtained by optimizing (18) with respect to $\alpha_{\mathcal{R},m}, \alpha^*_{\mathcal{R},m}, \alpha_{I,m}\,, \alpha^*_{I,m}$ and then substituting into (16).

Therefore, and after training phase, frequency responses at all subcarriers in each OFDM symbol can be obtained by SVR interpolation

$$\widehat{H}(k) = \sum_{m=0}^{N_P-1} \psi_m \, K(P_m, k) + b \tag{20}$$

for $k = 1, \cdots, N$. Note that, the obtained subset of Lagrange multipliers which are nonzero will provide with a sparse solution (and they represent the support vectors). Equation (20) will be used in the estimation step to predict the channel for the data symbols. As usual in the context of SVM framework, the free parameters of the kernel and the cost function represent limited freedom for the user and have to be fixed manually after gaining some a priori knowledge of the problem, or by using some validation set of observations [8].

5 Simulation results

In this part of our analysis, we compare the proposed algorithms (ANN with Backpropagation SCG algorithm and nonlinear complex RBF-based SVR algorithm) with the LS, Decision Feedback and perfect estimation based on Bit Error Rate (BER) and Mean Square Error (MSE) curves. We will analyze the

performance of our algorithms in terms of robustness against fading joint with nonlinear noise, and complexity.

We consider the channel impulse response of the time varying multipath fading channel model which can be written as

$$h(\tau,t)=\sum_{l=0}^{L-1}h_l(t)\,\delta(t-\tau_l) \tag{21}$$

where $h_l(t)$ is the impulse response representing the complex attenuation of the l^{th} path, τ_l is the random delay of the l^{th} path and L is the number of multipath replicas. The specification parameters of an extended vehicular A model (EVA) for LTE DL system with the excess tap delay and the relative power for each path of the channel are shown in table 1. These parameters are defined by 3GPP standard [15].

Table 1. Extended Vehicular A model (EVA) [15].

Excess tap delay [ns]	Relative power [dB]
0	0.0
30	-1.5
150	-1.4
310	-3.6
370	-0.6
710	-9.1
1090	-7.0
1730	-12.0
2510	-16.9

Figure 3 presents the variations in time and in frequency of the channel frequency response for a mobile speed equal to 350 Km/h.

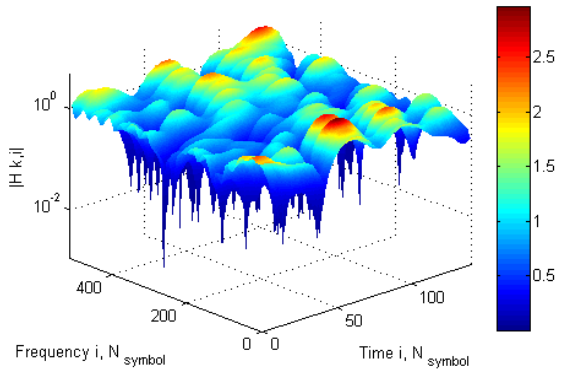


Figure 3: Variation in time and in frequency for a mobile speed at 350 Km/h.

In order to demonstrate the effectiveness of the presented techniques and evaluate the performance in the presence of nonlinear impulsive noise under high mobility conditions (350Km/h), two objective criteria are used: the signal-to-noise ratio (SNR) and signal-to-impulse ratio (SIR).

The expressions of SNR and SIR are given by [8]

$$SNR_{dB} = 10log_{10}\left(\frac{E\{|y(n) - w(n) - i(n)|^2\}}{\sigma_w^2}\right) \quad (22)$$

and

$$SIR_{dB} = 10log_{10}\left(\frac{E\{|y(n) - w(n) - i(n)|^2\}}{\sigma_{BG}^2}\right) \quad (23)$$

Then, we simulate the OFDM LTE DL system with parameters presented in Table 2. The proposed algorithms estimate a number of OFDM symbols in the range of 1400 symbols, corresponding to 10 radio frames LTE. Note that, the LTE radio frame duration is 10 ms [16], which is divided into 10 subframes. Each subframe is further divided into two slots, each of 0.5ms duration, as presented in Figure 4.

Table 3. Parameters of simulations [16], [17] and [18].

Parameters	Specifications
OFDM system	LTE/Downlink
Constellation	16-QAM
Mobile Speed (Km/h)	350
T_s (µs)	72
f_c (GHz)	2.15
δf (KHz)	15
B (MHz)	5
Size of DFT/IDFT	512
Number of paths	9

The variation of BER and MSE as a function of SNR in the presence of AWGN noise for a mobile speed at 350 Km/h is shown in Figs. 5 (a) and (b). The complex RBF-based SVR method slightly outperforms ANN with Backpropagation SCG algorithm and noticeably outperforms LS and DF methods. For example, we can see a gain of 10 dB over the DF method. We can also see that the SVR and ANN performances are close to the perfect channel knowledge estimation compared to others estimation techniques. These results demonstrate the advantage of the nonlinear complex SVR and its ability to adapt to the channel variations, providing a better channel estimation which gives an improvement of the service quality in LTE DL. MSE confirms the results obtained for BER and shows that LS suffers from a high MSE, however, complex SVR and ANN have low MSE.

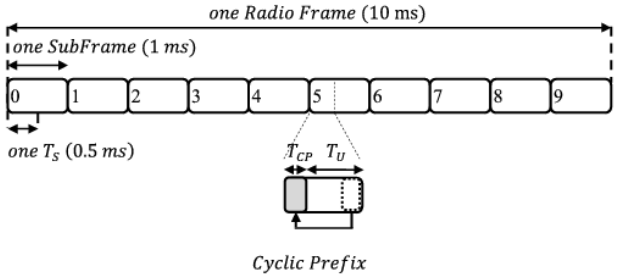


Figure 4: LTE frame structure

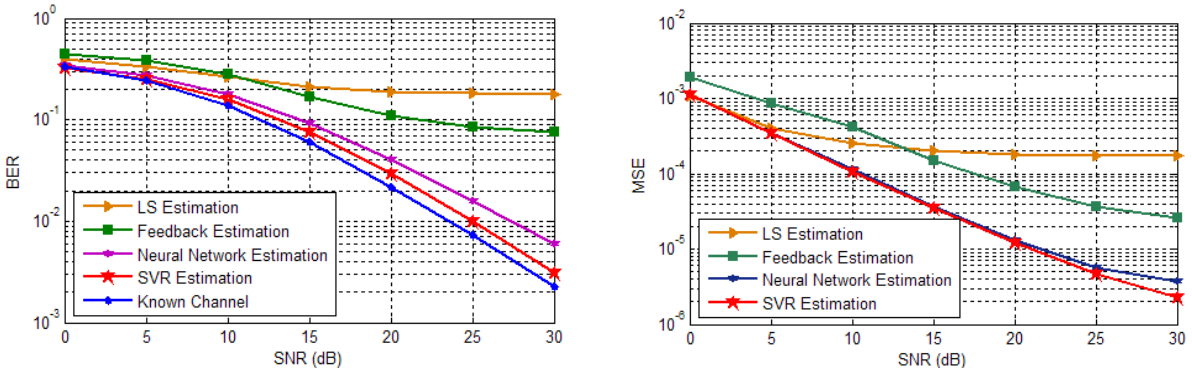


Figure 5: (a) BER and (b) MSE vs. SNR for a mobile speed at 350 Km/h without impulsive noise.

Figs. 6 (a) and (b) show the performance of LS, DF, ANN and complex SVR estimation techniques in the presence of AWGN noise and nonlinear impulsive noise with *SIR* = -5dB and *p* = .05 for a mobile speed at 350 Km/h. A poor performance is exhibited by LS and DF for all noise levels and good performance is observed with complex SVR which still track the estimation with perfect channel and also outperforms ANN. The MSE performance among these techniques ranges from best to the worst as follows: complex RBF-SVR based technique; SCG-ANN based technique, DF and LS.

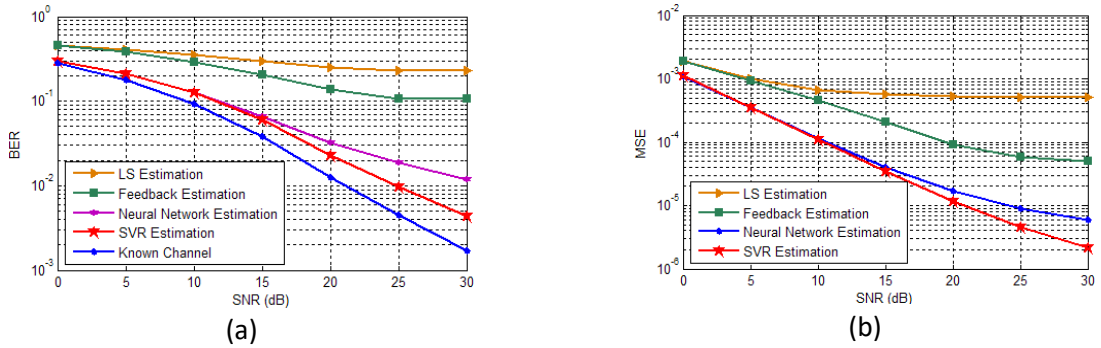


Figure 6: (a) BER and (b) MSE vs. SNR for a mobile speed at 350 Km/h with SIR = -5 dB and p = .05.

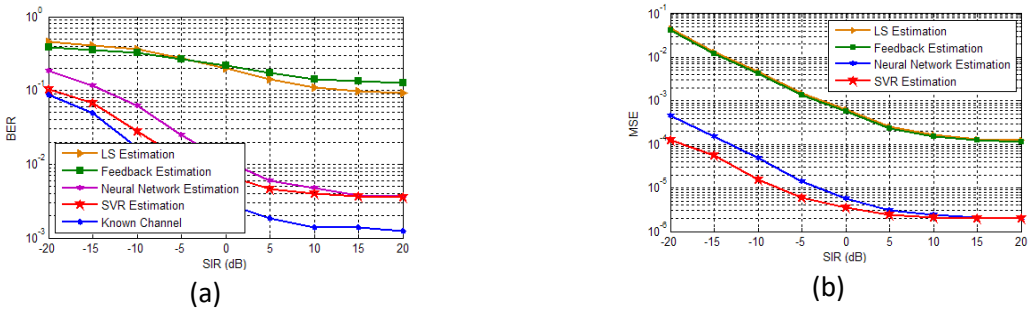
Figs. 7 (a) and (b) show the BER and MSE performances of LS, DF, ANN and complex SVR techniques in the presence of non-Gaussian impulsive noise with *p* = .05 for *SNR* = 30 dB as a function of SIR which is ranged from -20 to 20 dB. These figures confirm that nonlinear complex SVR algorithm performs better than LS, DF and also ANN algorithms in high-mobility environments presenting high levels of impulsive noise (SIR<0 dB), which proves that the RBF-SVR based approach is powerful in the nonlinear environments.

6 Conclusion

In this paper, we analyzed the performance of the scaled conjugate gradient backpropagation ANN and the complex RBF-based SVR algorithms for the channel estimation in the LTE DL system. These methods are based on two steps: the learning step where each method tries to adapt to the channel variations and constructs the regression model and the estimation step where the channel frequency response will be estimated.

The proposed methods where applied to vehicular A channel model according to 3GPP specifications in the presence of nonlinear impulsive noise interfering with OFDM pilot symbols in high-mobility environment. The simulation results clearly show that the nonlinear complex RBF-based SVR method

produces a good performance when compared to LS, Decision Feedback and ANN. The obtained results are very promising for improving the service quality in the LTE DL system. We still work on adopting this technique as channel estimator in LTE-Advanced systems.



(a) (b)

Figure 7: (a) BER and (b) MSE vs. SIR for a mobile speed at 350 Km/h with SNR = 30 dB and p = .05.

REFERENCES

[1]. Dahlman, E., S. Parkvall, J. Skold and P. Berning, *3G Evolution-HSPA and LTE for mobile broadband*. 2nd edition 2008, New York: vol. Academic.

[2]. Colieri, S., M. Ergen, A. Puri and A. Bahai. *A study of channel estimation in OFDM systems*. In Proceedings of the IEEE 56th Vehicular Technology Conference, 2002, vol. 2: p. 894–898.

[3]. Colieri, S., M. Ergen, A. Puri and A. Bahai. *Channel estimation techniques based on pilot arrangement in OFDM systems*. IEEE Transactions on Broadcasting, 2002, vol. 48, no. 3: p. 223–229.

[4]. Patra, J. C., R. N. Pal, R. Baliarsingh and G. Panda. *Nonlinear channel equalization for QAM signal constellation using artificial neural networks*. IEEE Transactions on Systems, Man, and Cybernetics, 1999, vol. 29, no. 2, p. 254–262.

[5]. Naveed, A., I. M. Qureshi, T. A. Cheema, and A. Jalil. *Blind equalization and estimation of channel using artificial neural network*, 8th International Multitopic Conference, INMIC, 2004, p. 184–190.

[6]. Fernández-Getino García, M. J., J. M. Páez-Borrallo, and S. Zazo. *DFT-based channel estimation in 2D-pilot-symbol-aided OFDM wireless systems*. IEEE Vehicular Technology Conf., 2001, vol. 2, p. 815–819.

[7]. Sliskovic, M. *Signal processing algorithm for OFDM channel with impulse noise*. IEEE conf. on Electronics, Circuits and Systems, 2000, p. 222–225.

[8]. Rojo-Álvarez, J. L., C. Figuera-Pozuelo, C. E. Martínez-Cruz, G. Camps-Valls, F. Alonso-Atienza, M. Martínez-Ramón. *Nonuniform interpolation of noisy signals using support vector machines*. IEEE Trans. Signal process., 2007, vol. 55, no.48, p. 4116–4126.

[9]. Nanping, L., Y. Yuan, X. Kewen, and Z. Zhiwei. *Study on channel estimation technology in OFDM system*. IEEE Computer Society Conf., 2009, p. 773–776.

[10]. Çiikli, C., A. T. Özsahin, A. C. Yapici. *Artificial neural network channel estimation based on Levenberg-Marquardt for OFDM systems*. Wireless Pers. Commun., 2009, p. 221–229.

[11]. Charrada, A and A. Samet. *Estimation of highly selective channels for OFDM system by complex least squares support vector machines. Int. J. Electron. Commun. (AEÜ), 2012, vol. 66, p. 687-692.*

[12]. Nanping, L., Y. Yuan, X. Kewen and Z. Zhiwei. *Study on channel estimation technology in OFDM system*. IEEE Computer Society Conf., 2009, p. 773–776.

[13]. Charrada, A and A. Samet. *Nonlinear Complex LS-SVM for Highly Selective OFDM Channel with Impulse Noise. 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), 2012, p. 696-700.*

[14]. Fernández-Getino García, M. J., J. L. Rojo-Álvarez, F. Alonso-Atienza, and M. Martínez-Ramón. *Support vector machines for robust channel estimation in OFDM*. IEEE signal process. J., 2006, vol. 13, no. 7.

[15]. 3rd Generation Partnership Project. Technical Specification Group Radio Access Network: evolved Universal Terrestrial Radio Access (UTRA): Base Station (BS) radio transmission and reception. *TS 36.104*, September 2009, V8.7.0.

[16]. 3rd Generation Partnership Project. Technical Specification Group Radio Access Network: evolved Universal Terrestrial Radio Access (UTRA): Physical Channels and Modulation layer. *TS 36.211*, September 2009, V8.8.0.

[17]. 3rd Generation Partnership Project. Technical Specification Group Radio Access Network: Physical layer aspects for evolved Universal Terrestrial Radio Access (UTRA). *TR 25.814*, September 2006, V7.1.0.

[18]. 3rd Generation Partnership Project. Technical Specification Group Radio Access Network: evolved Universal Terrestrial Radio Access (UTRA): Physical layer procedures. *TS 36.213*, September 2009, V8.8.0,

SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
ISSN 2054-7390

Application of Genetic Algorithms Coupled with Neural Networks to Optimization of Reinforced Concrete Footings

[1]Jiin-Po Yeh and [2]Shu-Yu Yeh
Department of Civil and Ecological Engineering, I-Shou University, Kaohsiung, Taiwan
[1]jpyeh@isu.edu.tw, [2]abcd781031@gmail.com

ABSTRACT

This paper first applies genetic algorithms to optimally design reinforced concrete isolated footings subjected to concentric loading. Based on the ACI Building Code, constraints are built by considering wide-beam and punching shears, bending moment, allowable soil pressure, the development length for deformed bars and clear distance between deformed bars. Design variables consist of the width, length and thickness of the footing and the number of bars in the long and short directions, all of which are discrete. The objective function is to minimize the cost of steel and concrete in the footing. There are totally 144 cases of reinforced concrete isolated footings considered. The optimal results are randomly divided into three groups for the use of neural networks: training data, validation data and testing data. Two kinds of artificial neural networks are employed in this paper: two-layer feedforward backpropagation networks and radial basis networks. Linear regression analysis between the network outputs and targets of the testing data is performed to judge the accuracy of the neural networks. Numerical results show that the feedforward backpropagation network is very effective and has high accuracy with the correlation coefficients and the slope of the regression line being close to one and the y-intercept close to zero. Besides, it is better than the radial basis networks and needs much fewer neurons in the hidden layer.

Keywords: Reinforced Concrete Isolated Footings; Genetic Algorithms; Feedforward Backpropagation Networks; Radial Basis Networks.

1 Introduction

Genetic algorithms are a heuristic search that is based on natural selection and natural genetics. It was inspired by the evolution theory of "survival of the fittest," which can solve both constrained and unconstrained optimization problems according to the "natural selection." The constraints used in genetic algorithms can be in the form of linear equality or inequality with bounds on the optimization variables. The concept of genetic algorithms was formally introduced in 1970s by Professor John Holland at the University of Michigan, who in 1975 published the ground-breaking book "Adaptation in Natural and Artificial System" [1] that led to many important discoveries. In 1989, Goldberg described in more detail the theory of genetic algorithms and its applications [2]. From then on, the continuing price/performance improvements of computational systems have made genetic algorithms more attractive and popular. Genetic algorithms have successfully been applied to many fields like engineering, economics, chemistry,

DOI: 10.14738/tmlai.44.2150
Publication Date: 20th August, 2016
URL: http://dx.doi.org/10.14738/tmlai.44.2150

manufacturing, mathematics, physics and so on. In the civil engineering, there are a lot of applications, such as optimal design of reinforced concrete beams [3], optimal design of planar and space structures [4], multiobjective optimization of trusses [5], reliability analysis of structures [6], global optimization of grillages [7], global optimization of trusses with a modified genetic algorithm [8], optimization of pile groups using hybrid genetic algorithms [9] and optimization of grid shell topology and nodal positions [10].

Artificial neural networks are a computational tool based on the properties of biological neural systems, which have been considered to be simplified models of neural processing in the brain. The artificial neural network was originated by McCulloch and Pitts in 1943 [11], who claimed that neurons with binary inputs and a step-threshold activation function were analogous to first order systems. In 1986, Rumelhart *et al*. [12] proposed the theory of parallel distributed processing and developed the most famous learning algorithm in ANN-backpropagation, which uses a gradient descent technique to propagate error through a network to adjust the weights in an attempt to reach the global error minimum, marking a milestone in the current artificial neural networks. Since then, a huge proliferation in the ANN methodologies has been taking place. In particular there are many applications to the civil engineering, such as structural optimization [13-15], damage identification of structural elements [16], frame optimization [17], traffic sign classification [18], optimal design of continuous reinforced concrete beams [19], etc.

Owning to the abilities of genetic algorithms to deal with highly nonlinear constraints and neural networks to build very complicated nonlinear relationships between inputs and outputs, this paper combines these two techniques to optimally design the reinforced concrete isolated footings. Based on the provisions of the ACI Building Code Requirements for Structural Concrete and Commentary [20], the constraints of genetic algorithms are constructed, considering the wide-beam and punching shears, bending moment, allowable soil pressure and the development length for deformed bars. The design variables are the effective depth, width and length of the footing, the areas of bending reinforcements in the long and short directions; the object is to find the minimum cost of concrete and steel.

2 Discrete Optimization

Most optimization approaches were focused on and developed for continuous variables. However, the variables are usually discreet for design problems. Genetic algorithms are simple and extremely capable in solving discrete optimization problems. Hence, this paper adopts genetic algorithms provided by the MATLAB Global Optimization Toolbox [21] to optimally design reinforced concrete beams with discrete variables. There are three major components in the operation of genetic algorithms: (1) creating a random initial population of designs (individuals); (2) combining the individuals in the population in order to produce better individuals; (3) obtaining a new generation of designs and going to the next step. Each individual is real-coded in this paper, which is composed of the design variables. To create the new population, the algorithms performs the following steps: (1) Score each individual of the current population by computing its fitness value; (2) Scale the raw fitness scores to convert them into a more usable range of values; (3) Select individuals, called parents, based on their fitness. The lower the value of the fitness function, the more opportunity it has to be selected; (4) Choose some elites from the current population that have lower fitness function values. These elite individuals are just passed to the next population; (5) Produce children from the parents. Children are produced either by making random changes to a single parent—mutation—or by combining the vector entries of a pair of parents—crossover; (6) Replace the current population with the crossover and mutation children and elites to form the next

generation. The algorithm stops when one of the stopping criteria is met, such as the number of generation, the weighted average change in the fitness function value over some generations less than a specified tolerance, no improvement in the best fitness value for an interval of time, etc.

The optimization problem of the reinforced concrete isolated footing is constituted as follows:

Minimize the fitness function $f(\boldsymbol{x})$

such that

$$C_i(\boldsymbol{x}) \leq 0, \quad i=1,\ldots, m$$

$$Ci(\boldsymbol{x})=0, \quad i=m+1,\ldots, mt \tag{1}$$

$$\boldsymbol{LB} \leq \boldsymbol{x} \leq \mathbf{UB}$$

where $C_i(\boldsymbol{x})$ represents the nonlinear inequality and equality constraints, m is the number of nonlinear inequality constraints, mt is the number of nonlinear constraints, $f(\boldsymbol{x})$ is the total cost of concrete and tension steels, and ***LB*** and ***UB*** are the vectors of lower and upper bounds of design variables, respectively. The Global Optimization Toolbox based on MATLAB uses the augmented Lagrangian genetic algorithm [22, 23] to solve nonlinear constraint problems with bounds. A subproblem is formulated by combining the fitness function and nonlinear constraint functions using the Lagrangian and the penalty parameters. A sequence of such optimization problems are approximately minimized using the genetic algorithm such that the bounds are satisfied. A subproblem formulation is defined as

$$\Phi(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{s}, \rho) = f(\boldsymbol{x}) - \sum_{i=1}^{m} \lambda_i s_i \log(s_i - C_i(\boldsymbol{x})) + \sum_{i=m+1}^{mt} \lambda_i C_i(\boldsymbol{x}) + \frac{\rho}{2} \sum_{i=m+1}^{mt} C_i(\boldsymbol{x})^2 \tag{2}$$

where the components λ_i of the vector $\boldsymbol{\lambda}$ are nonnegative and known as Lagrange multiplier estimates. The elements s_i of the vector $\boldsymbol{s}$ are nonnegative shifts, and ρ is the positive penalty parameter. The algorithm begins by using initial values for the parameters. Parameters s_i and λ_i are updated based on the value of $C_i(\mathbf{x})$. The genetic algorithm minimizes a sequence of the subproblem, which is an approximation of the original problem. When the subproblem is minimized to a required accuracy, the Lagrangian multiplier estimates are updated, or the penalty parameter is increased by a penalty factor. These steps are repeated until the stopping criteria of the genetic algorithm are met.

3 Artificial Intelligence

An artificial neural network is an analytical system that addresses problems without explicit solutions or whose solutions are very difficult to explicitly formulate. The neural network is composed of some computational units, called neurons, which are highly interconnected. Every interconnection has strength, called weight, which is represented by a number. The basic capability of neural networks is to learn patterns from a large number of examples by adjusting the weights of each neuron. The learning can be supervised or unsupervised. In this paper, the Neural Network Toolbox based on MATLAB is employed [24] and two kinds of artificial neural networks are used: the feedforward backpropagation and the radial basis. Both of them are supervised neural networks, which are briefly illustrated as follows.

3.1 The Feedforward Backpropagation Network

The most commonly used neural network is the feedforward neural network with the backpropagation learning algorithm. The network discussed in this paper has two layers: one hidden layer and one output layer, whose structure is shown in Figure 1. The transfer function of the single hidden layer is the tan-sigmoid function defined by

$$a_i = f(n_i) = \frac{e^{n_i} - e^{-n_i}}{e^{n_i} + e^{-n_i}} \quad , \quad i = 1,2,3,\ldots,k \tag{3}$$

where k is the number of the artificial neurons, $n_i = w_{i1}P_1 + w_{i2}P_2 + \ldots + w_{iR}P_R + b_i$, P_1, P_2,…P_R are the inputs, R is the number of input elements, w_{i1} , w_{i2} ,…, w_{iR} are the weights connecting the input vector and the ith neuron, and b_i is the bias of the ith neuron in the hidden layer. The output

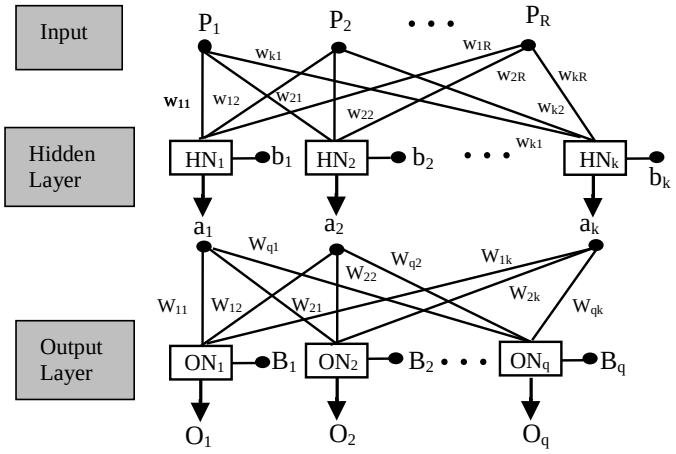


Figure 1 Two-layer feedforward backpropagation neural network with multiple outputs.

layer uses the linear transfer function defined by

$$Q_i = f(N_i) = N_i \quad , \quad i = 1,\ 2,\ldots,q \tag{4}$$

where $N_i = W_{i1}a_1 + W_{i2}a_2 + \ldots\ldots + W_{ik}a_k + B_i$, $W_{i1}, W_{i2}, \ldots, W_{is}$ are the weights connecting the neurons of the hidden layer and the ith neuron of the output layer, and B_i is the bias of the ith output neuron.

There are many variations of the backpropagation algorithm aiming to minimize the network performance function, i.e., the mean square error between the network outputs and the targets defined by

$$Minimize \quad MSE = \frac{1}{m}\sum_{j=1}^{m} (t_j - a_j)^2 \tag{5}$$

where t_j and a_j are the jth target and network output, respectively. Among many training functions, the Levenberg-Marquardt algorithm [25, 26] was chosen to minimize the network performance function. The formula to update the weights and biases is given by

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k - [\boldsymbol{J}^T\boldsymbol{J} + \mu\ \boldsymbol{I}]^{-1}\boldsymbol{J}^T\boldsymbol{e} \tag{6}$$

where $\boldsymbol{X}_k$ is a vector of current weights and biases, $\boldsymbol{J}$ is the Jacobian matrix of the first derivative of the error vector $\boldsymbol{e}$ between the network outputs and target outputs with respect to the weights and biases, $\boldsymbol{I}$ is a unit matrix and μ is a parameter. If μ→0, Eq. (6) can be simplified as

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k - [\boldsymbol{J}^T\boldsymbol{J}]^{-1}\boldsymbol{J}^T\boldsymbol{e} \approx \boldsymbol{X}_k - \boldsymbol{H}^{-1}\boldsymbol{J}^T\boldsymbol{e} \tag{7}$$

which is the quasi-Newton's method with the approximate Hessian matrix *H* [33]; if μ→∞, Eq. (6) turns out to be

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k - \mu^{-1}\boldsymbol{J}^T\boldsymbol{e} \tag{8}$$

which is the gradient descent method with the learning rate μ^{-1} [24]. Therefore, this algorithm interpolates between the quasi-Newton's algorithm and the gradient descent method. If a tentative step increases the performance function, the parameter μ will be increased, causing this algorithm to act like the gradient descent method, while it shifts toward Newton's method if the reduction of the performance function is successful, i.e., the parameter μ will be decreased. In this way, the performance function will always be reduced at each iteration of the algorithm. To improve the network generalization, the error on the validation set is monitored simultaneously during the training process. When the network begins to overfit the training data, the error on the validation set typically begins to rise. Once the validation error increases for a specified number of iterations (The default value set by MATLAB is six), the training terminates and the weights and biases at the minimum of validation error are returned.

The number of neurons required in the hidden layer is usually unknown beforehand. Bayesian regularization [27] provides a measure of how many network parameters (weights and biases) are being effectively used by the network. According to this effective number of parameters, the number of neurons required in the hidden layer of the two-layer neural network can be estimated by the following equation

$$(Rk+k)+(kq+q)= Num \tag{9}$$

where *R* and *q* are the number of elements in the input and output vectors, respectively, *k* is the number of neurons to be determined in the hidden layer, and *Num* is the effective number of parameters found by the Bayesian regularization implemented by the function *trainbr* in MATLAB.

3.2 The Radial Basis Network

For comparison with the feedforward backpropagation network, this paper uses another network, the radial basis network that has two layers: the radial basis layer and output layer. The transfer function in the artificial radial basis neuron is the radial basis function defined by

$$a = radbas(n) = e^{-n^2} \tag{10}$$

as shown in Figure 2, where n= $\|\boldsymbol{w} - \boldsymbol{P}\|\ b$ is the vector distance (Euclidean distance) between the weight vector w and the input vector P multiplied by the bias b. As the distance between w and P decreases, the output increases. Thus the radial basis function acts as a detector that produces 1.0 whenever the input is identical to the weight vector. Each bias in the radial basis layer is set to be 0.8326/SPREAD, which causes radial basis function to output 0.5 when $\|\boldsymbol{w} - \boldsymbol{P}\|$ = +/- SPREAD. The larger the constant SPREAD is,

the smoother the radial basis function will be. The constant needs to be large enough so that several radial basis neurons respond to overlapping region of the input space and have fairly large output at any given instant, but not so large that all the neurons respond in essentially same manner [24]. Different *SPREADs* are usually tried to find the best value for a given problem. There are two functions in MATLAB to design the radial basis network: *newrb* and *newrbe.*

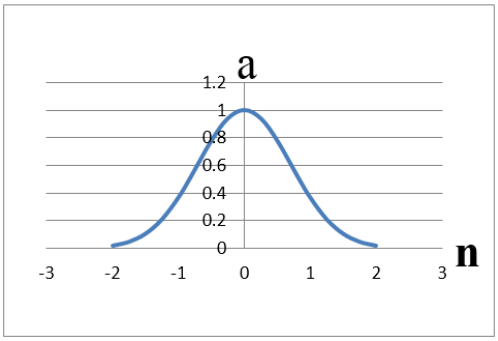


Figure 2 Radial basis function

3.2.1 The Design Function newrb

The function *newrb* iteratively creates one radial basis neuron at a time. At each iteration, the input vector that results in lowering the network error is used to create a radial basis neuron. Neurons are added to the network until the sum-squared error falls beneath an error goal or maximum number of neurons has been reached.

3.2.2 The Design Function newrbe

The function *newrbe* can produces a network with zero error on the training vectors. It creates as many radial basis neurons as there are input vectors, and each neuron acts as detector for a different input vector. The drawback to *newrbe* is that it produces a network with as many hidden neurons as there are input vectors. For this reason, it does not return an acceptable solution when many input vectors are needed to properly define a network, as is typically the case.

To make the above-mentioned neural networks more efficient, it is often useful to scale inputs and targets so that they will always fall within a specific range. For example, the following formula

$$y = 2(\frac{x - min}{max - min}) - 1 \tag{11}$$

is used in this paper to scale inputs and targets, where x is the original value, y is the scaled value, and *max* and *min* are the maximum and minimum of inputs or targets, respectively. Eq. (11) produces inputs and targets in the range [-1, 1]. To evaluate the performance of the trained network, this paper makes use of a regression analysis between the network outputs and the corresponding targets.

4 Design of Reinforced Concrete Isolated Footings

The reinforced concrete isolated footings considered in this paper are loaded concentrically, as shown in Figure 3, with width B, length L and thickness h. The dead and live loads transmitted by the column are denoted by P_D and P_L, respectively. The column size is $a \times b$. A variety of reinforced concrete isolated footings are optimally designed by the genetic algorithm. The objective function is to minimize the total

cost of the concrete and the tension reinforcements in the long and short directions of the rectangular isolated footing. All the constraints required to design the isolated footing comply with the ultimate-strength design of ACI 318-08 Code, considering wide-beam and punching shears, bending moment and the development length for deformed bars. The units of force and length in the following formulas are kgf (=9.81N) and cm, respectively.

4.1 Shear

The shear strength of the isolated footing in the vicinity of column reactions is governed by the more severe of the following two conditions:

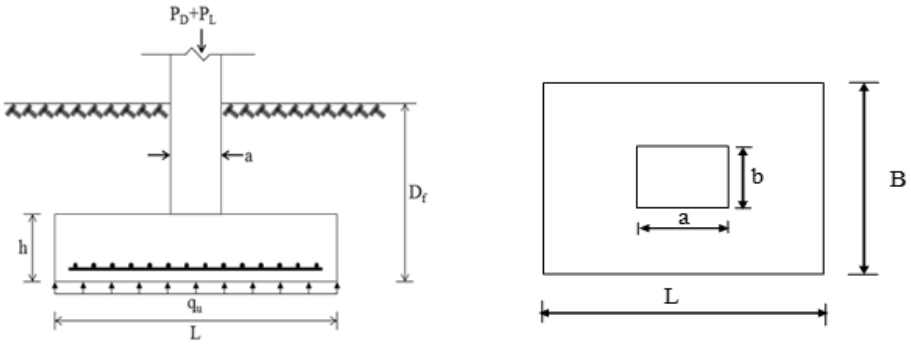


Figure 3 The isolated footing: (a) elevation and (b) plan.

4.1.1 Wide-beam Shear

The critical section is assumed to extend in a plane across the entire width and lies at a distance *d* from the face of the column, as shown in Figure 4(a). The nominal shear strength of this section is

$$V_{c1} = 0.53\sqrt{f_c'}Bd \tag{12}$$

or

$$V_{c2} = 0.53\sqrt{f_c'}Ld \tag{13}$$

where *d* is the effective depth of the footing. The constraints for wide-beam shear are

$$V_{u1} = q_u\left(\frac{L-a}{2} - d\right)B \le \phi V_{c1} \tag{14}$$

and

$$V_{u2} = q_u\left(\frac{B-b}{2} - d\right)L \le \phi V_{c2} \tag{15}$$

where $q_u = (1.2P_D+1.6P_L)/(BL)$ is the factored net soil pressure and ϕ =0.75 is the strength reduction factor for shear.

4.1.2 Punching Shear

The critical section occurs at a distance *d*/2 from the face of the column, as shown in Figure 4 (b). The maximum allowable nominal shear strength is the smallest of the following three equations

$$V_c = (0.53 + \frac{1.06}{\beta_c})\sqrt{f_c'}b_0 d \ ,$$
$$V_c = (0.53 + \frac{0.265\alpha_s d}{b_0})\sqrt{f_c'}b_0 d \quad (16)$$
$$V_c = 1.06\sqrt{f_c'}b_0 d$$

where β_c = long side *a*/short *b* of the column, b_0 = perimeter of the critical section *ijkl* and α_s = 40, 30 and 20 for interior, edge and corner columns, respectively. In this paper, interior columns are considered; therefore, α_s = 40. Similarly, the constraint for the punching shear is

$$V_u = q_u[BL - (a+d)(b+d)] \le \phi V_{c,\min} \quad (17)$$

where $V_{c,min}$ is the smallest of Eqs. (16). The area to be considered for factored shear V_u is equal to the total area of the footing less the area *ijkl*.

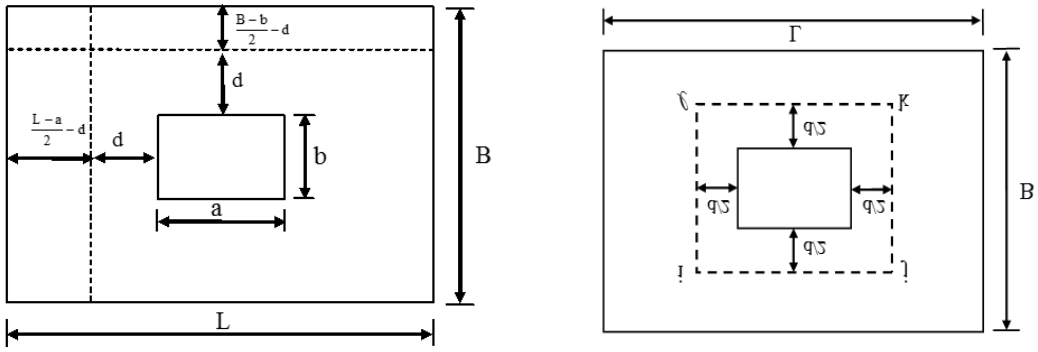


Figure 4 Critical sections: (a) wide-beam shear and (b) punching shear

4.2 Bending Moment

Let A_b be the cross-sectional area of the reinforcement steel and N_L and N_B be the number of steels in the long and short directions of the footing, respectively. The critical section for bending moment is at the face of the column and the constraints are

$$M_{uL} = q_u \frac{B}{2}(\frac{L-a}{2})^2 \le \phi_{mL} N_L A_b f_y (d - \frac{N_L A_b f_y}{2(0.85) f_c' B}) \quad (18)$$

and

$$M_{uB} = q_u \frac{L}{2}(\frac{B-b}{2})^2 \le \phi_{mB} N_B A_b f_y (d - \frac{N_B A_b f_y}{2(0.85) f_c' L}) \quad (19)$$

where ϕ_{mL} and ϕ_{mB} are the strength reduction factor for moment. Let ε_t be the tensile strain of the reinforcement, then

$$0.65 \le \phi_{mL} \text{ or } \phi_{mB} = 0.65 + 0.25\frac{\varepsilon_t - \varepsilon_y}{0.005 - \varepsilon_y} \le 0.9 \tag{20}$$

To prevent sudden failure with little or no warning when the beam cracks or fails in a brittle manner, the ACI code limits the minimum and maximum amount of steel to be

$$A_{sL,\min} \le N_L A_b \le A_{sL,\max} = \frac{0.85 f_c' \beta B d}{f_y}(\frac{3}{7}) \tag{21}$$

and

$$A_{sB,\min} \le N_B A_b \le A_{sB,\max} = \frac{0.85 f_c' \beta L d}{f_y}(\frac{3}{7}) \tag{22}$$

where β is the stress block depth factor,

$$A_{sL,\min} \ge \max\ (\frac{0.8\sqrt{f_c'}}{f_y} Bd,\ \frac{14Bd}{f_y}) \tag{23}$$

and

$$A_{sB,\min} \ge \max\ (\frac{0.8\sqrt{f_c'}}{f_y} Ld,\ \frac{14Ld}{f_y}) \tag{24}$$

The formula for $A_{sL,max}$ in Eq. (21) or $A_{sB,max}$ in Eq. (22) is derived based on the requirement that the tensile strain must be greater than or equal to 0.004. In addition, both the steel ratios $N_L A_b/(Bh)$ and $N_B A_b/(Lh)$ must exceed the minimum value required for temperature and shrinkage: 0.0018 for grade 60 deformed bars and 0.002 for grade 40 or 50 deformed bars.

4.3 Allowable Soil Pressure

Suppose that the allowable soil pressure under the base of the footing is q_a. The gross soil pressure must not exceed the allowable soil pressure, that is,

$$\frac{P_D + P_L}{BL} + hw_c + \gamma_s(D_f - h) \le q_a \tag{25}$$

where D_f is the distance from the base to the ground surface, as shown in Figure 3, w_c is the weight of concrete and γ_s is the unit weight of soil over the footing.

4.4 Development of Reinforcement

According to the ACI Code, the equation for the development length of bars in tension is

$$\ell_d = \frac{0.15 d_b f_y \psi_t \psi_e \lambda}{\sqrt{f_c'}} \tag{26}$$

for No. 6 and smaller bars with clear spacing not less than $2d_b$ and clear cover not less than d_b, where d_b is the bar diameter, and ψ_t and ψ_e are the bar location and coating factors, respectively. In this paper ψ_t and

ψ_e are assumed to be 1.0. For normal weight concrete, λ=1. The critical section for development-length of the bars in tension is the same as the critical section in flexure, that is, at the face of the column. Hence,

$$0.5(L\text{-}a) \text{ - concrete cover} \geq \ell_d \tag{27}$$

and

$$0.5(B\text{-}b) \text{ - concrete cover} \geq \ell_d \tag{28}$$

The equation for the development length of bars in compression is

$$\ell_{dc} = \max\ (\frac{0.075 d_b f_y}{\sqrt{f_c'}},\ 0.0043 d_b f_y) \tag{29}$$

The dowel bars stressed to f_y are required to transfer the axial compression force in the column into the footing, as shown in Figure 5; hence, there should be minimum extension of the dowels

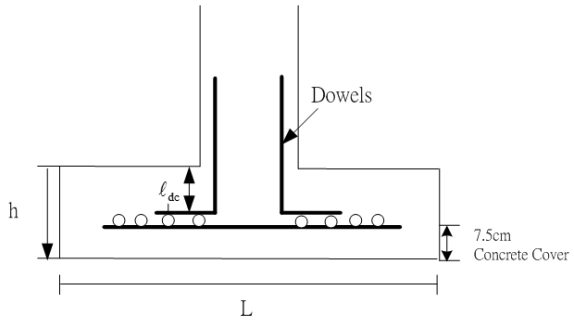


Figure 5 Dowels that transfer the column load to the footing slab.into the footing. Therefore, the thickness *h* of the footing must satisfy the following constraints:

$$h - \text{concrete cover} - 2d_b\,(\text{footing bars}) - d_b\,(\text{dowels}) \geq \ell_{dc} \tag{30}$$

4.5 Reinforcement Distribution and Clear Distance

In the short direction of the footing, a central band of width *B* shall contain the major portion of flexural reinforcement according to the formula $2N_B/(L/B+1)$ but rounded up to the nearest integer and uniformly distributed along the band width. The remainder is also uniformly distributed outside the central band. The reinforcement in the long direction is uniformly distributed across the entire width of the footing. The clear distance *s* between individual steel bars both in the long and short directions must satisfy

$$Min\ (3h,\ 45\ cm) \geq s \geq Max\ (2d_b,\ 2.5\ cm) \tag{31}$$

5 Numerical Results

The given design conditions for the isolated footing are as follows: the concentric dead load P_D, the live load P_L applied to the column, the allowable soil pressure q_a at the base of the footing, the distance from the footing bottom to the ground surface D_f, the compressive strength of concrete f_c', and the yield stress of steel f_y. In addition, the column size is assumed to be 0.45 m×0.45 m, the unit weight of soil over the footing is 2000 kgf/m^3, and the unit weight of concrete is 2400 kgf/m^3. In Taiwan, the unit prices of concrete and steel are 1900 NT$/m^3 and 18.4 NT$/kgf, respectively. The concrete cover for the

reinforcement of the footing is assumed to be 7.5 cm. The optimal results obtained from genetic algorithms consist of the minimum cost of the footing, the thickness h, width B and length L of the footing, and the number of steel bars in the long and short directions, N_L and N_B, respectively. Based on the often-used materials and customs in Taiwan, this paper selects two kinds of yield strength f_y of the tension reinforcement: 2800 kgf/cm^2 (40 ksi) and 4200 kgf/cm^2 (60 ksi) as well as two kinds of compressive strength f'_c of the concrete: 210 kgf/cm^2 (3000 psi) and 280 kgf/cm^2 (4000 psi). There are three kinds of q_a: 25 ton/m^2, 30 ton/m^2 and 35 ton/m^2; three kinds of D_f: 1.0 m, 1.5 m and 2.0 m; four kinds of dead load P_D: 60 ton, 80 ton, 100 ton and 120 ton as well as four kinds of P_L: 40 ton, 60 ton, 80 ton and 100 ton. There are totally144 kinds of footings being designed depending on the combination of the six given conditions. For the purpose of training, monitoring and testing the neural networks, the optimal data are divided into 3 groups: 100 training sets (70%), 22 validation sets (15%) and 22 testing sets (15%). The fitness function is the total cost in New Taiwan Dollars of the footing reinforcement and concrete. All the constraints are built according to the formulas discussed in Sec. 4.

5.1 Genetic Algorithms

This paper adopts the MATLAB global optimization toolbox to carry out genetic algorithms. To run genetic algorithms of the MATLAB software, some parameters need to be selected beforehand. After a number of trials, here are the values used in this paper: The population size is set to be 100, crossover rate 0.8, and elite number 6. Furthermore, all the individuals are encoded as integers; "Rank" is used as the scaling function that scales the fitness values based on the rank of each individual; "Roulette" is the selection function to choose parents for the next generation; "Two-point crossover" is used as the crossover method to form a new child for the next generation; The "Adaptive Feasible Function" is chosen as the mutation function to make small random changes in the individuals and ensure that linear constraints and bounds are satisfied. The steel bars used in the footing can usually range from No. 5 to No. 10. In order to decide the appropriate size of steel bars to be used in this paper, different sizes of reinforcement bars are tried for a variety of random combinations of f_y, f'_c, P_D, P_L, q_a and D_f. The results show that No. 5 bars always lead to the minimum cost. Therefore, No. 5 steel bars are used in each direction of the footing and as dowels that transfer the column load to the footing slab. Taken as an example, Table 1 shows the results for the case of f_y =2800 kgf/cm^2, f'_c =210 kgf/cm^2, P_D = 120 ton, P_L= 100 ton, q_a =35 ton/m^2 and D_f =1 m.

Table 1 The optimal results for different sizes of reinforcement bars for the case of f_y =2800 kgf/cm^2, f'_c =210 kgf/cm^2, P_D = 120 ton, P_L= 100 ton, q_a =35 ton/m^2 and D_f =1 m.

Size of bars	h(m)	B (m)	L(m)	N_B	N_L	cost(NT$10^3)
5	0.65	2.27	2.96	42	32	14.343
6	0.66	2.31	2.91	29	23	14.573
7	0.66	2.31	2.91	22	17	14.624
8	0.66	2.27	2.96	17	13	14.655
9	0.67	2.27	2.96	13	10	14.683
10	0.67	2.47	2.82	10	9	15.383

5.2 Feedforward Backpropagation Networks

The inputs of the artificial neural network consist of six elements: f_y, f'_c, P_D, P_L, q_a and D_f, and the targets also have six components: the minimum cost, h, B, L, N_B, N_L and the cost. To make the network more efficient, all the data are normalized by using Eq. (11) or the function *mapminmax* introduced in the MATLAB software for neural network. The function *trainbr* is first applied to find the number of effective parameters required for the network, which is found to be 118.9, as shown in Figure 6; therefore the number of neurons required in the hidden layer is 8.685 according to Eq. (9), which is then rounded up to 9. After the number of neurons needed in the hidden layer is determined, a very efficient training function *trainlm*, i.e., the Levenberg-Marquardt algorithm, is used to train the network. While the network is being trained, the training and validation data are both put into the network. During the training process, the network error for training data will decrease, while the network error for the validation data decreases first but increases later on. To avoid overfitting the data, the training will terminate when the network error of the validation data increases continuously for a number of epochs whose default value is six set by the MATLAB software. The training process is shown in Figure 7, where there are totally 23 epochs but the weights and biases at the epoch 17 are taken as those of the trained network. The value of the performance function at the epoch 17 is 0.0019623. After the normalized data are reversed to the original scale, the six network outputs and targets for the testing data are presented in Figs. 8-13. The 22 sets of testing data containing inputs and targets as well as outputs are shown in Tables 2(a) and 2(b), respectively. If more neurons in the hidden layer are used, for example, 12 neurons, the accuracy of the network can only improve a little bit, not significantly. Tables 3 shows the linear regression analysis results for the testing data when the hidden layer has 9 and 12 neurons, where parameters m, b and r stand for the slope of the straight line, the intercept with the vertical axis and correlation coefficient, respectively. Table 3 indicates that the networks with 9 neurons and 12 neurons in the hidden layer have almost the same accuracy. Based on the Figs. 8-13 and Table 3, the performance of the network is considered quite good on the whole, because the parameters m and r are close to one and parameter b also close to zero.

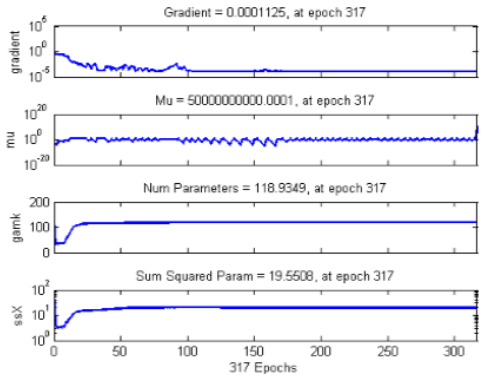


Figure 6 The number of effective parameters in the network obtained from the training function trainbr

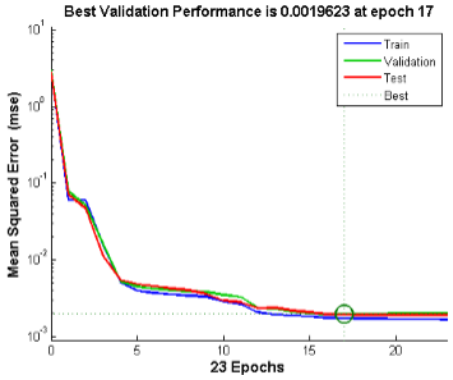


Figure 7 The training process for the feedforward backpropagation neural network with 9 neurons in the hidden layer

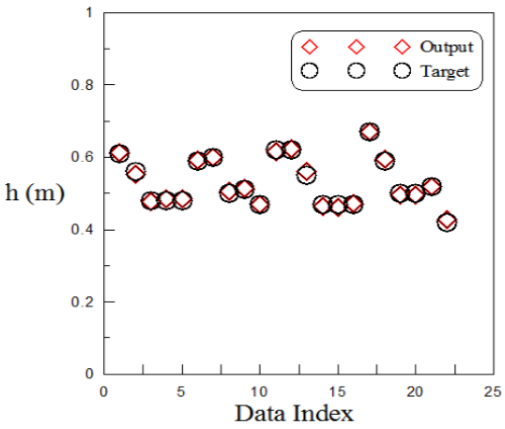


Figure 8 Outputs and targets of the footing thickness h for the testing data with 9 neurons in the hidden layer of the feedforward backpropagation network

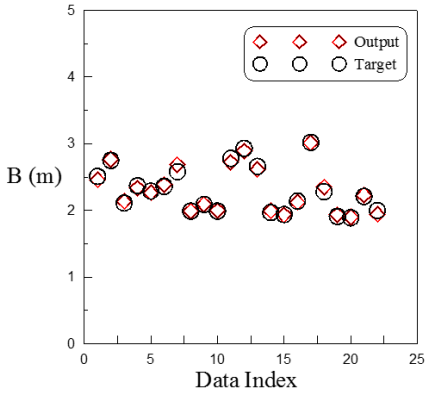


Figure 9 Outputs and targets of the footing width *B* for the testing data with 9 neurons in the hidden layer of the feedforward backpropagation network

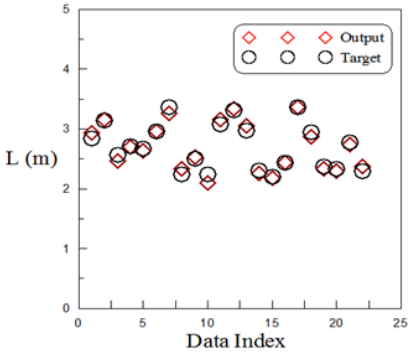


Figure 10 Outputs and targets of the footing Length *L* for the testing data with 9 neurons in the hidden layer of the feedforward backpropagation network

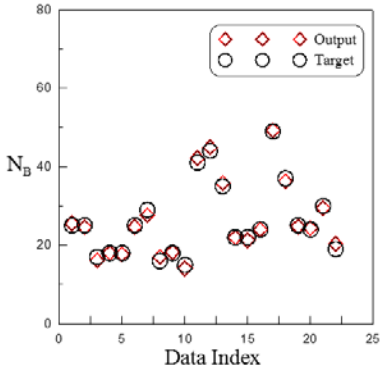


Figure 11 Outputs and targets of the number of steel bars in the short direction for the testing data with 9 neurons in the hidden layer of the feedforward backpropagation network

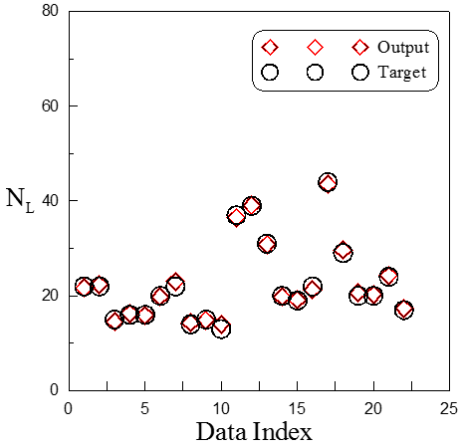


Figure 12 Outputs and targets of the number of steel bars in the long direction for the testing data with 9 neurons in the hidden layer of the feedforward backpropagation network

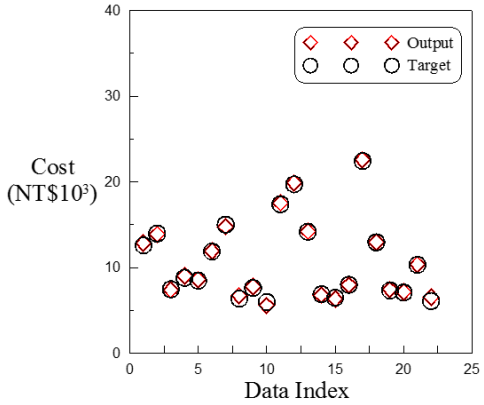


Figure 13 Outputs and targets of the total cost for the testing data with 9 neurons in the hidden layer of the feedforward backpropagation network.

Table 2(a) Inputs and targets of the testing data with 9 neurons in the hidden layer of the feedforward backpropagation network.

Data index	Inputs						Targets					
	f_y (kg/cm^2)	f_c' (kg/cm^2)	P_D (ton)	P_L (ton)	q_a (ton/m^2)	D_f (m)	h (cm)	B (cm)	L (cm)	N_B	N_L	Cost (NT$10^3)
1	4200	280	120	100	35	2	61	251	285	25	22	12.585
2	4200	280	100	80	25	2	56	275	315	25	22	13.972
3	4200	280	80	60	30	2	48	211	257	17	15	7.478
4	4200	280	80	60	25	1.5	48	237	271	18	16	8.812
5	4200	280	80	60	25	1	48	229	268	18	16	8.469
6	4200	210	100	80	30	2	59	236	296	25	20	11.876
7	4200	210	100	80	25	2	60	258	336	29	22	15.000
8	4200	210	80	60	35	1.5	50	199	224	16	14	6.377
9	4200	210	80	60	30	1.5	51	209	250	18	15	7.624
10	4200	210	60	40	35	1.5	47	199	224	15	13	5.982
11	2800	280	120	100	30	2	62	278	308	41	37	17.397
12	2800	280	120	100	25	1	62	293	331	44	39	19.723
13	2800	280	100	80	25	1	55	266	297	35	31	14.189
14	2800	280	80	60	35	2	47	197	231	22	20	6.897
15	2800	280	80	60	35	1	47	194	220	22	19	6.476
16	2800	280	80	60	30	1.5	47	214	244	24	22	7.990
17	2800	210	120	100	25	1.5	67	302	336	49	44	22.422
18	2800	210	100	80	30	1.5	59	228	295	37	29	12.957
19	2800	210	80	60	35	2	50	191	238	25	20	7.333
20	2800	210	80	60	35	1.5	50	189	233	24	20	7.090
21	2800	210	80	60	25	1	52	221	278	30	24	10.319
22	2800	210	60	40	25	1.5	42	200	229	19	17	6.106

Table 2(b) Network outputs of the testing data with 9 neurons in the hidden layer of the feedforward backpropagation network

Data index	Network outputs					
	h (cm)	B (cm)	L (cm)	N_B	N_L	Cost (NT$10^3)
1	61.3	246.1	293.2	25.5	21.6	12.860
2	55.4	276.5	315.4	24.5	22.4	13.893
3	47.8	213.5	247.1	16.2	14.6	7.411
4	48.6	233.4	269.9	18.0	16.3	9.038
5	48.5	226.7	263.2	17.7	15.9	8.591
6	59.1	238.6	295.9	24.6	19.8	11.933
7	60.1	268.6	325.8	27.6	23.0	14.771
8	50.5	198.8	233.8	17.0	14.4	6.745
9	51.4	209.0	253.3	18.2	14.9	7.804
10	46.9	199.8	209.5	14.1	14.0	5.582
11	61.6	272.2	315.5	42.2	36.5	17.563
12	62.3	288.8	334.6	45.0	39.1	19.888
13	56.0	261.3	305.4	35.8	30.8	14.250
14	46.5	199.4	225.4	21.9	19.8	6.793
15	46.3	193.2	218.4	21.3	19.2	6.338
16	47.3	212.3	243.9	24.0	21.3	7.960
17	67.0	301.3	336.5	49.3	43.5	22.571
18	59.4	234.8	287.6	36.3	29.7	12.965
19	49.6	193.5	233.5	24.7	20.7	7.431
20	49.5	190.2	230.2	24.3	20.3	7.184
21	51.9	223.0	273.9	29.5	24.3	10.355
22	42.6	194.2	237.2	20.3	17.3	6.541

Table 3 The linear regression analysis of targets and outputs for the testing data with 9 and 12 neurons in the hidden layer of the feedforward backpropagation network.

No. of neurons in the hidden layer	Outputs (targets)	m (slope)	b (y-intercept)	r (correlation coefficient)
9	h	1.0008	0.0044	0.9975
	b	0.9767	-0.0029	0.9933
	L	1.0168	-0.0069	0.9861
	N_B	1.0191	0.0048	0.9973
	N_L	0.9825	-0.00001	0.9985
	Cost	1.0031	0.0073	0.9992
12	h	0.9755	-0.001	0.9983
	b	0.9687	-0.0001	0.9908
	L	1.0113	0.0089	0.9906
	N_B	0.9787	-0.0036	0.9966
	N_L	0.9965	-0.0001	0.9986
	Cost	1.0019	0.0025	0.9996

5.3 Radial Basis Networks

There are two kinds of design functions for the radial basis network: *newrb* and *newrbe*. The *newrbe* produces as many number of radial basis neurons as the input vectors; therefore, it has zero error between the network output and target, while the *newrb* creates one neuron at a time until the preset mean square error between the network outputs and the targets or the number of epochs are reached. Because the radial basis network does not require the validation data, only training and testing data are considered. For comparison, these two sets of data are intended to be the same as those of feedward backpropagaton with 9 neurons in the hidden layer.

5.3.1 The Function *newrb*

The mean square error between the network outputs and targets is set to be 0.0019623, which is the same as the result of the feedforward backpropagation. Let the parameter *SPREAD* change from 1.0 to 2.0 and train the network for each case. By observing the regression analysis of the network outputs and targets, all of them show very good performance, while the network with *SPREAD*=1.6 is a little bit better than other values; therefore, *SPREAD*=1.6 is selected for the radial basis layer. From the training process, it is found that 44 epochs are required for the mean square error to fall beneath the goal of 0.0019623; therefore, 44 radial basis neurons are created for the network. The testing data are then substituted into the trained network. The linear regression analysis of the six network outputs and targets is shown in Table 4, where the correlation coefficients are between 0.9801 and 0.9939. Tables 3 and 4 indicate that the performance of the radial basis network is a little inferior to the feedforward backpropagation network.

Table 4 The linear regression analysis of the outputs and targets for the testing data using *newrb* with SPREAD=1.6.

Outputs (Targets) / Parameters	h	B	L	N_B	N_L	Cost
m	0.9971	0.9488	1.0194	0.9775	0.9676	0.9747
b	-0.029	-0.0303	-0.0225	-0.0297	-0.0301	-0.0269
r	0.9913	0.9826	0.9801	0.9899	0.9930	0.9939

5.3.2 The Function *newrbe*

The default value for the mean square error is set to be zero. In order to compare with the *newrb* function, let *SPREAD* =1.6. Because the network can achieve zero error, all the training data will lead to the exactly fitting regression model with the parameters $m = r = 1$ and $b = 0$. Then, substitute the testing data into the trained network. The linear regression analysis of the six network outputs and targets for the testing data is shown in Table 5, where the correlation coefficients are between 0.9656 and 0.9983.

Table 5 The linear regression analysis of the outputs and targets for the testing data using *newrbe* with SPREAD=1.6.

Outputs (Targets) / Parameters	h	B	L	N_B	N_L	Cost
m	1.0406	0.9772	1.1108	1.0522	0.9823	1.0433
b	-0.0042	-0.0073	0.0241	0.0182	-0.013	0.0152
r	0.9942	0.9819	0.9656	0.9903	0.9944	0.9983

6 Conclusions

This paper first applies the genetic algorithm to engage in the optimal design of the reinforced concrete isolated footings. There are 144 different isolated footings being designed, the results of which are used as the training, validation and testing data of the artificial neural networks. From the numerical results, the principal conclusions may be summarized as follows:

(1) According to the effective number of parameters obtained from the *trainbr* function, this paper only uses 9 neurons in the hidden layer for the feedforward backpropagation network. Even if more neurons are used in the hidden layer, the accuracy does not improve significantly. The correlation coefficients between the network outputs and targets range from 0.9861 to 0.9992 for testing data. In addition, the slope of the regression line is close to 1 and the intercept with the vertical axis close to zero. The results suggest very good performance of the feedforward backpropagation network.

(2) The parameter *SPREAD*=1.6 is the most suitable value for the *newrb* design function of the radial basis network. The correlation coefficients between the network outputs and targets range from 0.9801 to 0.9939 for testing data. On the whole, its performance is a little inferior to the feedforward backpropagation network. Besides it needs more radial basis neurons in the hidden layer. If the *newrbe* function is used, the results of the regression analysis are similar to *newrb*, although it can reach zero error during the training process.

(3) After the artificial neural network is trained, it can serve as a model to optimally design the reinforced concrete isolated footings effectively and efficiently. For practical purposes, the outputs of the neural network can be rounded up to the nearest whole number.

REFERENCES

[1] Holland, J. H. (1975), *Adaptation in natural and artificial systems*, The University of Michigan Press, Ann Arbor, MI, USA.

[2] Goldberg, D. E. (1989), *Genetic algorithms in search, optimization and machine learning*, Addison Wesley, Reading, MA, USA.

[3] Coello, C. C., Hernandez, F. S., and Farrera, F. A. (1997), "Optimal design of reinforced concrete beams using genetic algorithms," *Expert Systems with Applications*, Vol. 12, No. 1, pp. 101-108.

[4] Erbatur, F, Hasancebi, O., Tutuncu, I, and Kilic, H. (2000), "Optimal design of planar and space structures with genetic algorithms," *Computers and Structures*, Vol. 75, No. 2, pp. 209-224.

[5] Coello, C. A. and Christiansen, A. D. (2000), "Multiobjective optimization of trusses using genetic algorithms," *Computers and Structures*, Vol. 75, pp. 647-660.

[6] Cheng, J and Li, Q. S. (2008), "Reliability analysis of structures using artificial neural network based genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, Vol. 197, No. 45. pp. 3742-3750.

[7] Belevičius, R. and Šešok, D. (2008), "Global optimization of grillages using genetic algorithms," *Mechanika*, Nr. 6(74), pp. 38-44.

[8] Šešok, D. and Belevičius, R. (2008), "Global optimization of trusses with a modified genetic algorithm," *Journal of Civil Engineering Management*, Vol. 14, No. 3, pp. 147-154.

[9] Chan, C. M., Zhang, L. M. and Jenny, T. N. (2009), "Optimization of pile groups using hybrid genetic algorithms, "*Journal of Geotechnical and Geoenvironmental Engineering*, Vol. 135, Issue 4, pp. 497-505.

[10] Richardson, J. N., Adriaenssens, S., Coelho, R. F. and Bouillard, P. (2013), "Coupled form-finding and grid optimization approach for single layer grid shells," *Engineering Structures*, Vol. 52, pp. 230-239.

[11] McCulloch, W.S. and Pitts, W. (1943), "A logical calculus of ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, Vol. 5, pp. 115-133.

[12] Rumelhart, D. E., McClelland, J. L. and the PDP Research Group (1986), *Parallel distributed processing: explorations in the microstructure of cognition. volume 1: foundations*, MIT Press, Cambridge, MA, USA.

[13] Cheng, J. and Li, Q. S. (2009), "A hybrid artificial neural network method with uniform design for structural optimization," *Computational Mechanics*, Vol. 44, No. 1, pp. 61-71.

[14] Möller, O., Foschi, R. O., Quiroz, L. M., and Rubinstein, M. (2009), "Structural optimization for performance-based design in earthquake engineering: applications of neural networks," *Structural Safety*, Vol. 31, No. 6, pp. 490–499.

[15] Gholizadeh, S. and Salajegheh, E. (2010), "Optimal design of structures for earthquake loading by self organizing radial basis function neural networks," *Advances in Structural Engineering*, Vol. 13, No. 2, pp. 339-356.

[16] Nazarko, P. and Ziemiański, L. (2011), "Application of artificial neural networks in the damage identification of structural elements," *Computer Assisted Mechanics and Engineering Sciences*, Vol. 18, No. 3, pp. 175–189.

[17] Meon, M. S., Anuar, M. A., Ramli, M. H. M., Kuntjoro, W., and Muhammad, Z. (2012), "Frame optimization using neural network", *International Journal Advanced Science Engineering Information Technology*, Vol. 2, No. 1, pp. 28-33.

[18] Ciresan, D. C., Meier, U., Masci, J. and Schmidhuber, J. (2012), "Multi-column deep neural network for traffic sign classification," *Neural Networks*, Vol. 32, pp. 333-338.

[19] Yeh, J.-P. and Yang, R.-P. (2015), "Optimal Design of Continuous Reinforced Concrete Beams Using Neural Networks," *Transactions on Machine Learning and Artificial Intelligence*, Vol. 3, No. 4, pp. 1-16.

[20] ACI (2008), *Building code requirements for structural concrete* (*ACI 318-08*) *and commentary* (*ACI 318R-08*), American Concrete Institute, Farminton Hills, MI, USA.

[21] The MathWorks (2015), *Global optimization toolbox: user's guide*, The MathWorks, Inc., Natick, MA, USA.

[22] Conn, A. R., Gould, N. I. M., and Toint, Ph. L. (1991), "A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds," *SIAM Journal on Numerical Analysis*, Vol. 28, No. 2, pp. 545-572.

[23] Conn, A. R., Gould, N. I. M., and Toint, Ph. L. (1997), "A globally convergent augmented Lagrangian barrier algorithm for optimization with general inequality constraints and simple bounds," *Mathematics of Computation*, Vol. 66, No. 217, pp. 261-288.

[24] Demuth, H., Beale, M. and Hagan, M. (2009), *Neural network toolbox 6: user's guide*, The MathWorks, Inc., Natick, MA, USA.

[25] Hagan, M. T., Demuth, H. B. and Beale, M. H. (1996), *Neural network design*, PWS Publishing, Boston, MA, USA.

[26] Pujol, J (2007), "The solution of nonlinear inverse problems and the Levenberg-Marquardt method". *Geophysics*, Vol. 72, No. 4, W1–W16.

[27] MacKay, D. J. C. (1992), "Bayesian interpolation," *Neural Computation*, Vol. 4, No. 3, pp. 415-447.