Journal of Quality Measurement and Analysis *JQMA* 20(1) 2024, 187-204
e-ISSN: 2600-8602 *http://www.ukm.my/jqma*
https://doi.org/10.17576/jqma.2001.2024.14

PREDICTION ANALYSIS OF COVID-19 IN SELANGOR BY USING BACKPROPAGATION ALGORITHM WITH CONJUGATE GRADIENT METHOD

(Analisis Ramalan COVID-19 di Selangor dengan Menggunakan Algoritma Perambatan Balik dengan Kaedah Kecerunan Konjugat)

NOOR AMIRAH BINTI AJMAL KHAN & SITI MAHANI MARJUGI*

ABSTRACT

COVID-19 is a disease that can spread rapidly among individuals in a population. COVID-19 is a kind of coronavirus. Patients infected with COVID-19 can lead to death, especially people with lung difficulties or a weakened immune system. COVID-19 is transmitted to humans through contact, coughing, sneezing, or close contact. As a result, using previous COVID-19 data in Selangor, an artificial neural network (ANN) is used as an effective future prediction method. Backpropagation is a form of artificial neural network (ANN) algorithm that may be used to resolve issues in prediction analysis. Predictive analysis is utilized to take some earlier action to produce more effective outcomes, which is minimizing the number of COVID-19 patients, to avoid the COVID-19 disease spreading drastically and becoming worse. However, total performance is determined by the optimization approach discovered throughout the training phase. The Fletcher-Reeves approach can improve the efficiency of the backpropagation algorithm by having a faster convergence rate than other methods such as the scaled conjugate gradient method. Based on this theory, this study developed a backpropagation neural network using the conjugate gradient method to create the prediction analysis of COVID-19 patients in Selangor. The result of the experiment will show how many COVID-19 patients can be obtained from the mean square error (MSE) of the test data at various learning rates. The effective learning rate is determined by applying the MSE to both the test and training data, which results in the lowest number of MSE, which is the ideal option. From the output of testing data, the number of predicted confirmed COVID-19 patients can be determined.

Keywords: COVID-19; artificial neural network (ANN); Fletcher-Reeves; backpropagation algorithm; mean square error (MSE); training data; learning rates

ABSTRAK

COVID-19 merupakan penyakit yang boleh merebak dengan cepat dalam kalangan individu dalam sesuatu populasi. COVID-19 adalah sejenis coronavirus. Pesakit yang dijangkiti COVID-19 boleh menyebabkan kematian, terutamanya orang yang mengalami masalah paru-paru atau sistem imun yang lemah. COVID-19 berjangkit kepada manusia melalui sentuhan, batuk, bersin, atau sentuhan rapat. Hasilnya, menggunakan data COVID-19 sebelum ini di Selangor, rangkaian saraf tiruan (ANN) digunakan sebagai kaedah ramalan masa depan yang berkesan. Perambatan balik ialah satu bentuk algoritma rangkaian saraf tiruan (ANN) yang boleh digunakan untuk menyelesaikan isu dalam analisis ramalan. Analisis ramalan digunakan untuk mengambil beberapa tindakan lebih awal untuk menghasilkan hasil yang lebih berkesan, iaitu meminimumkan bilangan pesakit COVID-19, untuk mengelakkan penyakit COVID-19 merebak secara drastik dan menjadi lebih teruk. Walau bagaimanapun, jumlah prestasi ditentukan oleh pendekatan pengoptimuman yang ditemui sepanjang fasa latihan. Pendekatan Fletcher-Reeves boleh meningkatkan kecekapan algoritma perambatan belakang dengan mempunyai kadar penumpuan yang lebih cepat daripada kaedah lain seperti kaedah kecerunan konjugat berskala. Berdasarkan teori ini, kajian ini membangunkan rangkaian neural perambatan balik menggunakan kaedah kecerunan konjugat untuk mencipta analisis ramalan pesakit COVID-19 di Selangor. Hasil eksperimen akan menunjukkan berapa ramai pesakit

COVID-19 boleh diperoleh daripada ralat min kuasa dua (MSE) data ujian pada pelbagai kadar pembelajaran. Kadar pembelajaran yang berkesan ditentukan dengan menggunakan MSE pada kedua-dua data ujian dan latihan, yang menghasilkan bilangan MSE terendah, yang merupakan pilihan yang ideal. Daripada output data ujian, bilangan pesakit COVID-19 yang diramalkan yang disahkan boleh ditentukan.

Kata kunci: COVID-19; rangkaian saraf tiruan (ANN); Fletcher-Reeves; algoritma perambatan balik; ralat min kuasa dua (MSE); data latihan; kadar pembelajaran

1. Introduction

Newly discovered COVID-19 in Wuhan, China, in December 2019 caused an outbreak that spread throughout Asia, Africa, and the United States, as well as Taiwan and Malaysia; the virus was previously known as coronavirus or COVID-19. The development of an infectious disease that attacks the respiratory system left the globe in a state of disbelief. On 11 March 2020, the World Health Organization declared an epidemic of the disease in China. Patients who have contracted COVID-19 have described symptoms ranging from mild to severe. Within a week of virus exposure, symptoms usually appear 2 to 14 days later. Anyone can have moderate to severe symptoms. Symptoms often start with nonspecific symptoms such as fever, chronic cough, and lethargy. Several systems may be affected especially the lungs (cough, difficulty breathing, sore throat, runny nose, hematuria, and chest tightness), gastrointestinal tract (diarrhea, vomiting, and nausea), musculoskeletal system (muscle pain), and neurological (headache or confusion). Fever, cough, and shortness of breath are the most common signs and symptoms. About 15% of people have a fever, cough, and breathlessness (Thimm *et al.* 1996). As a precaution, those who test positive for COVID-19 must be quarantined for about two weeks. Patients who do not excrete during this time can transmit the disease to others around them through physical contact and respiratory droplets from the air they breathe.

To determine how many COVID-19 patients have been infected, several parameters must be considered, including mortality and recovery rates. The 14-day incubation period of the COVID-19 chain in the body is likely to alter the next day's count evaluation. Figures 1(a) and 1(b) show an increase in the number of COVID-19 cases in Malaysia and Selangor for the period of 1 January to 15 May 2022.

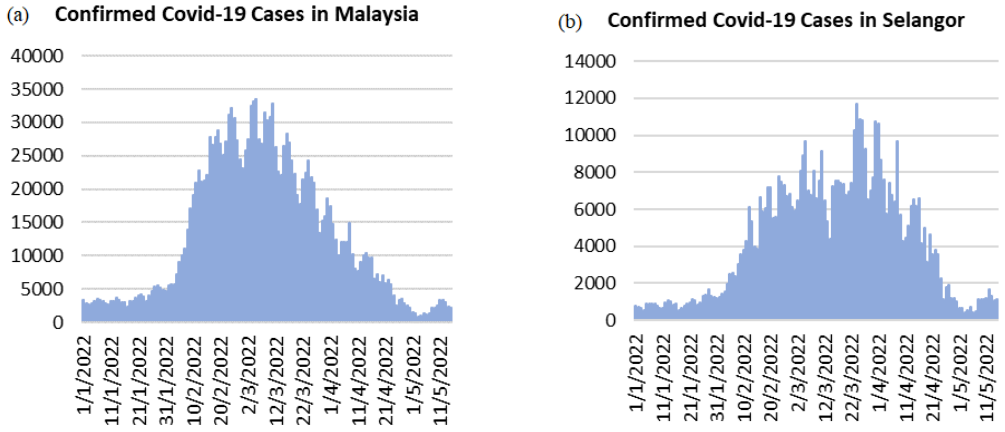

Figure 1: (a) The number of COVID-19 cases in Malaysia, (b) The number of COVID-19 cases in Selangor

To curb the transmission of COVID-19, government authorities have taken precautionary measures such as limiting urban community movement in a certain region, including Malaysia, known as the Movement Control Order (MCO). This guideline supports public officials in establishing a physical distance between persons to interrupt the chain of infection. Several strategies for predicting the spread of illnesses have been presented. During the MCO, all mass events and gatherings are banned, all business and premises those selling daily essentials are closed, and all educational institutions of all categories are shuttered (Chong & Żak 2013).

Regression is most often used to analyze time series data, although it is ineffective when dealing with aspects that are more complicated. Artificial neural network (ANN) outperforms regression because it is more adaptable and able to handle more complex and unexpected situations. Since backpropagation algorithm which algorithm that works inside artificial neural network takes a long time in the data prediction learning process, so it takes an optimization technique method that can improve the learning process. Optimization technique is the process where it trained the model iteratively that results in minimizing function evaluation. It is one of the most important tasks in machine learning to get better results. Machine learning models need to be optimized because it need to be compared the results in every iteration by changing the parameters in each step until reach the optimum results. The accurate model is with less error rate. The machine learning algorithm, backpropagation performs the backward pass to adjust the parameters of model and it aiming to minimize the mean squared error (MSE). Backpropagation also strengthen the connections within layers to achieve at a desired prediction. These are method that used inside backpropagation algorithm to optimize a model namely gradient descent and Fletcher-Reeves method. The Fletcher-Reeves (FR) method is used to optimize a backpropagation neural network (BPNN), which is discussed at the end of this study. The COVID-19 patients are predicted using backpropagation algorithm with Fletcher-Reeves method.

The ANN approach handles information processing systems. ANN is quite similar to the theoretical model of human neuron cognition induction. In ANN, many neurons communicate with each other. The information is sent from one neuron to the neuron that can receive it. Weight is the name given to this relationship in ANN. It is the kind of interactions that occur between each neuron in an ANN that define its architecture. Weights are also used to determine the relationship between neurons. Training data for the ANN layer may be directed by using the backpropagation approach, which is organized. Patients from the COVID-19 registry are used as training data. Training data comes from real COVID-19 patient's data in Selangor. Backpropagation algorithm is a well-known approach for tackling more complex problems and finding the optimal solution. The neural network architecture uses the backpropagation approach to develop a thorough knowledge of COVID-19 patients in order to accomplish the aim. When dealing with large amounts of data, the number of iterations required by the backpropagation algorithm increases dramatically. When using a backpropagation neural network, the conjugate gradient approach, Fletcher-Reeves method is used to boost the convergence rate much more efficiently compared to gradient descent.

Because of COVID-19 data is quite comprehensive, a specific time period was chosen to study the association between immunization and the Omicron. The number of incidents increased in March 2022 due to the increased infectiousness of the Omicron. An individual who had complete immunization and had the booster dose experienced a less severe version of the disease. The backpropagation algorithm with Fletcher-Reeves was utilized in this study to achieve a short-term strong empirical of COVID-19 instances.

1.1. *Research background*

It is common practice to apply gradient-based minimized the error techniques for training backpropagation networks. There are several layers of neural network that feed into each other in the backpropagation training method (Russell & Norvig 2010). Neural network weight known as backward error correction is added to a gradient descent optimum scheduling technique.

Neuron network training efficacy is a key study topic for future development, and various publications have already been written on this topic. In the beginning, the backpropagation algorithm was reworked. As a result, as detailed by Bishop (Hestenes & Stiefel 1952), many optimization techniques for enhancing the effectiveness of the error reduction process and, as a result, the efficiency of training were applied. The conjugate gradient approach for Hestenes and Stiefel (1952) and Huang's set of Quasi-Newton algorithms (Huang 1970) are two examples of how Fletcher and Powell (1963) and the Fletcher and Reeves (1964) may expand it. Gain variation in a nonlinear activation has recently been shown to accelerate the accumulation rate of the standard backwards-propagation approach (Eom *et al.* 2003; Maier & Dandy 1998; Thimm *et al.* 1996). A gain modification, however, has not been found to alter a training iteration's optimal gradient. Therefore, we will create and test a few keys by CG formulas to improve the algorithm's convergence rate. This study proposes a major change in search direction in order to greatly increase the training efficiency of almost all essential (well-known and built) optimization techniques. A considerable increase in convergence rates may be obtained independently of the optimization techniques by locally adjusting the search direction by the acquired value utilized in the input signal of the associated node.

1.2. *Problem statement*

Since backpropagation algorithm takes a long time in the data prediction learning process, so it takes a method that can improve the learning process. In order to face the hectic situations, the government and other functional bodies need the researchers help to predict the future number of COVID-19 patients in a short time and take some precautions to avoid the spread of the COVID-19 in community. Therefore, Fletcher-Reeves (FR) method is inserted into the backpropagation neural network (BPNN) in order to optimize the results as well as the time taken for the learning process.

1.3. *The proposed method*

Predicting the number of COVID-19 patients can be done using the backpropagation with conjugate gradient method. It is possible to predict the number of COVID-19 patients in Selangor during the next several months using the flowchart in Figure 2 (Anam *et al.* 2021) and numerous variables connected to the COVID-19 virus's transmission. Another advantage of the backpropagation algorithm is that the input components are based on characteristics that influence the distribution of COVID-19 patients. Patients who die, those who recover, and those who become ill 14 days earlier than expected are all variables in COVID-19's spread. The backpropagation algorithm's initial weights are generated at random. In other words, the backpropagation algorithm improves with each new set of experiment data. During the learning phase, the weights of the neural network are updated when the output comes closer to the target or actual value. The Fletcher-Reeves method may help reduce this error. Usually, a common technique is to use a cost function which measure the error between predicted and actual output. Then by using optimization technique, Fletcher-Reeves method, the model will adjust its parameter to minimize the cost function. During the network test, the term "e final weight" is utilized. Firstly, the error, e which is the difference between predicted and actual output that

acquire from output layer. Then, error is sent back to each neuron in backward direction from output layer to hidden layer. Therefore, the "e final weight" denotes gradient of error which is calculated with respect to each weight is obtained. The ultimate weight of the COVID-19 test subjects is used to predict the number of patients. The accuracy of the method is evaluated using test data from both the training and testing data. The following is the reasoning for the study: In the previous 14 days, the number of COVID-19 patients has changed owing to the deaths of patients, the recovery, and new positive cases.

The learning rate controls how quickly the model is adapted to the problem and range is between 0.0 and 1.0. Choosing the optimal learning rate can greatly improve the training of neural network and can prevent any odd behavior that may occur. For learning rate selection, a better approach is to start with large value of learning rate and gradually decrease the value. Therefore, it will optimize weights closer to the optimal value by changing it slightly. These known as learning rate decay and is shown to help with optimization and generalization. The higher learning rate can cause the model to converge too quickly and may lead to overshooting the optimal values while the lower learning rate can cause the process to get stuck. As a result, this study's learning methods were selected which are 0.001, 0.005, 0.01, 0.1, and 0.2 learning rates.

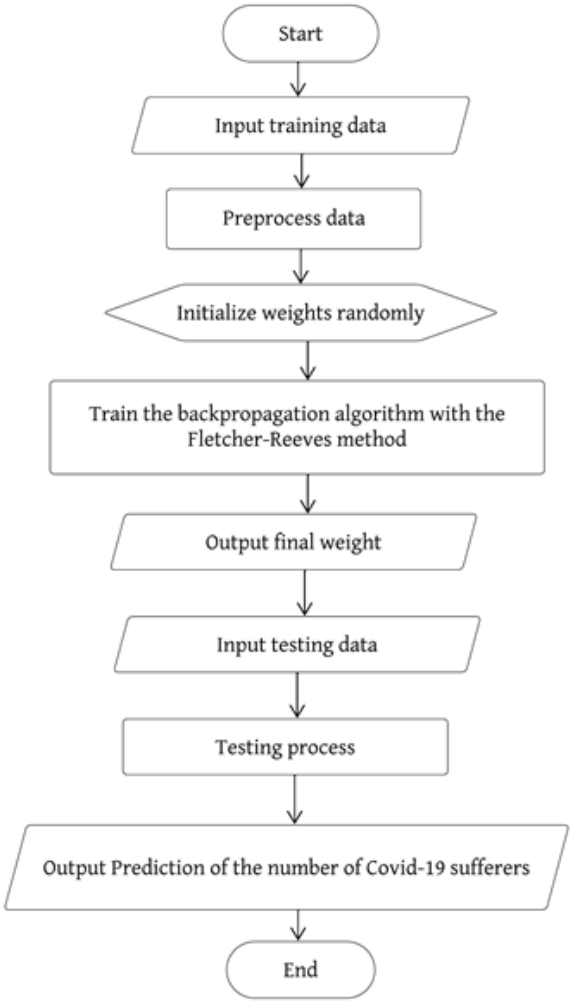

Figure 2: The flowchart system of Backpropagation algorithm with Fletcher-Reeves

2. Methodology

2.1. *Introduction*

There are detailed descriptions of the study datasets, optimization, the Fletcher-Reeves method, the backpropagation algorithm, and the suggested strategy. Among the concepts covered in this section are the Fletcher-Reeves method and the backpropagation algorithm, which analyze the experiment's dataset. At the conclusion of this section, the backpropagation neural network is explained in more depth using the Fletcher-Reeves method and backpropagation. With this method, the number of COVID-19 sufferers may be determined. These are the parameters that had been used in the methodology part, p (search direction), g (the gradient of the problem), α (learning rate), β_t (Fletcher-Reeves method), X_i (input units), z_j (hidden units), y_t (output units), v and w (weight) and δ_t (error information).

There are three layers in backpropagation such as input, hidden and output layer. The backpropagation algorithm can be summarized in the following steps. Firstly, forward pass where input data is fed into the neural network and it performs a series of mathematical operations, known as activation functions, to produce a prediction. Next, error calculation had used to determine the difference between the predicted output and actual output in which computed using a loss function. Then, backward pass, the algorithm calculates the gradient of the loss function with respect to each parameter in the network using the chain rule of calculus. It calculates how much each parameter contributes to the overall error. Subsequently, parameter update where the networks parameters (weights and biases) are updated using an optimization algorithm by moving opposite direction of the computed gradients. This step aims to minimize the loss function, making the predictions more accurate. At last, iterative process in which the first four steps are repeated for multiple iterations (epochs) over the entire training dataset. Each iteration fine-tunes the network's parameters to reduce the prediction errors gradually.

2.2. *Algorithm 1: The Fletcher–Reeves algorithm*

(1) Input the initial point x_0, terminating criterion, and maximum number of iterations, $t_{\max}$.

(2) Initialize $t = 0$.

(3) Calculate p_0 by using Eq. (1), the search direction. It is formulated as a negative of the gradient of the function,

$$p_0 = -g_0 = f(x_0). \tag{1}$$

(4) While $(\nabla f(x_t) > \varepsilon)$ or $(t \le t_{\max})$, do Calculate p_t by using

$$p_t = -g_t + \beta_t p_{t-1}. \tag{2}$$

Choose α, which minimizes $f(x_t + \alpha p_t)$, Calculate x_{t+1} by using

$$x_{t+1} = x_t + \alpha p_t. \tag{3}$$

where α is the learning rate.

Calculate β_t by using,

$$\beta_t = \frac{g_t^T g_t}{g_{t-1}^T g_{t-1}}. \tag{4}$$

where β_t is called the Fletcher-Reeves method.

(5) End While.

2.3. *Algorithm 2: Backpropagation algorithm*

(1) Initialize weights.

(2) While termination condition is false

(i) Feedforward:

For data processing, the hidden layer receives the input units $X_i, (i = 1, 2, 3, \ldots, n)$ which is obtained from the list signal. In order to get the output of a hidden unit $z_j, (j = 1, 2, 3, \ldots, m)$ the following Eq. (5) is used. The value is transfer to the next layer:

$$\begin{aligned} z_{in_j} &= v_{0j} + \sum_{i=1}^{n} x_i v_{ij}, \\ z_j &= f(z_{in_j}). \end{aligned} \tag{5}$$

The input has been weighted using the equation below, and each unit output $y_t, (t = 1, 2, 3, \ldots, p)$ adds it up:

$$y\_in_t = w_{0t} + \sum_{j=1}^{m} z_j w_{jt}. \tag{6}$$

By figuring out the activation function in the equation below, we can get the output signal:

$$y_t = f(y\_in_t). \tag{7}$$

(ii) Backpropagation of error:

Each output unit $(y_t, t = 1, 2, 3, \ldots, p)$ has a pattern that associated with the input pattern, and the equation below is used to figure out the error information:

$$\delta_t = (k_t - y_t) f'(y\_in_t). \tag{8}$$

In order to update the weights later, the error correction is computed,

$$\Delta w_{jt} = \alpha \delta_t z_j. \tag{9}$$

With Eq. (10) the bias correction is also computed, and δ_t is then passed on to the layer units after that:

$$\Delta w_{0t} = \alpha \delta_t. \tag{10}$$

The input delta is computed by multiplying the error data from the previous layer's units by the output weight and adding the result:

$$\delta\_in_j = \sum_{t=1}^{p} \delta_t w_{jt}. \tag{11}$$

The activation function's first derivative is determined. As a result, the error information is amplified by this factor,

$$\delta_j = \delta\_in_j f'(z\_in_j). \tag{12}$$

Following the error correction, the weights are updated using Eq. (13):

$$\Delta v_{ij} = \alpha \delta_j x_i. \tag{13}$$

Furthermore, the bias correction is calculated using the following equation:

$$\Delta v_{0j} = \alpha \delta_j. \tag{14}$$

(iii) Updating weights and biases:
Each output unit's weights and biases are updated using the formula in Eq. (15) to decrease the error:

$$w_{jt}^{new} = w_{jt}^{old} + \Delta w_{jt}. \tag{15}$$

Hence, each hidden unit's weights and biases are updated using this equation's formulation:

$$v_{ij}^{new} = v_{ij}^{old} + \Delta v_{ij}. \tag{16}$$

Check the stop condition in (2).

2.4. *Algorithm 3: Backpropagation algorithm with the Fletcher–Reeves method*

(1) Initialized $ik = 1$ and weight, v_{ij} and w_{jt}.

(2) Input $ik_{\max}$,

(3) While $ik < ik_{\max}$ do

(i) Feedforward

An input signal is sent to input units $X_i, (i = 1, 2, 3, \ldots, n)$ and it passed on to the hidden layer.

The Eq. (17) and Eq. (18) are used to calculate the output of each hidden unit $z_j, (j = 1, 2, 3, \ldots, m)$. The output a value will transfer to the next layer is ensured by this equation:

$$z\_in_j = v_{0j} + \sum_{i=1}^{16} x_i v_{ij}. \tag{17}$$

The activation function is used to compute the output signal. Therefore, the value able to be transferred to the next layer. The hidden layer is activated using the activation function, while the output layer is the log-sigmoid function, as shown in the following equation:

$$z_j = f(z_{in_j}) = \frac{1}{1 + e^{-z\_in_j}}. \tag{18}$$

The output unit (y) totals the weighted inputs, represented by

$$y_{in} = w_0 + \sum_{j=1}^{m} z_j w_j. \tag{19}$$

The activation function for calculating the output signal used is defined by

$$y = f(y_{in}) = \frac{1}{1 + e^{-y_{in}}}. \tag{20}$$

(ii) Backpropagation of error:

There are several output units $y_t, (t = 1, 2, 3, \ldots, p)$ each of which is connected to each other with the input patterns. Eq. (11) is used to determine the error information:

$$\delta = (k - k) f'(y_{in}). \tag{21}$$

All of the delta input from the previous layer's hidden units is added to the current layer's hidden units $z_j, (j = 1, 2, 3, \ldots, m)$:

$$\delta\_in_j = \delta w_j. \tag{22}$$

(iii) Updating weights and biases:

The following equation may be used to get p_{ik}'s value:

$$p_{ik} = -g_{ik} + \beta_{ik} p_{ik-1}. \tag{23}$$

The value of β_{ik} is obtained using the following equation:

$$\beta_{ik} = \frac{g_{ik}^T g_{ik}}{g_{ik-1}^T g_{ik-1}}. \tag{24}$$

The biases and weights in the output unit $(j = 1, 2, 3, ..., m)$ are updated with Eq. (5):

$$w_j^{ik+1} = w_j^{ik} + \alpha p_{ik}(\delta z_j). \tag{25}$$

The biases and weights $(i = 0, \ldots, 16)$ are updated for each hidden unit $Z_j, (j = 1, 2, 3, \ldots, m)$:

$$v_j^{ik+1} = v_j^{ik} + \alpha p_{ik}(\delta_j x_i). \tag{26}$$

Check the stop condition (3).

3. Results and Discussion

3.1. *Introduction*

The results of the study suggest that both the number of neurons in the hidden layer and the learning optimization technique both have an impact on the performance of backpropagation neural networks (BPNN). When the number of hidden neurons is large, the generalizability of the technique is diminished. This is known as an overfitting condition. Having an overly low or high learning rate will result in poor accuracy and learning failure. Backpropagation and backpropagation neural network (BPNN) with Fletcher-Reeves methods are compared using MSE in order to see how well its function lowers the value of MSE, then it is considered as the best method. The Fletcher–Reeves optimization method combined with a backpropagation neural network (BPNN) improves the predictions for future COVID-19 cases.

To implement the prediction system, the hardware used is a laptop with an AMD Athlon Gold 3150U processor with Radeon Graphics, 2.40 GHz, 8GB RAM, 256 SSD, programmed on matlab.mathworks.com.

3.2. *Normalization dataset*

The dataset is collected from COVIDNOW website. As shown in Table 1, the data before normalization includes cases active, cases dead and cases recovered with respect to the date.

After applying the normalization process in, as presented in Table 2, the data is transformed and the models uses the number of COVID-19 sufferers within the previous 14 days (xd_1 to xd_14), the number of deaths (xd_15), and the number of recoveries up to the previous day (xd_16), while the output from the network is the number of confirmed cases to date (target), but in order to achieve better results the data is normalized by dividing it by $2x10^5$ which is randomly picked, and that helps avoiding exploding gradient problem, it also helps the models converge faster, as if it's not normalized, the error the weights update with becomes huge and overshoots.

Table 1: COVID-19 dataset before normalization

NO.	DATE	CASES ACTIVE	CASES DEAD	CASES RECOVERED
1	1/1/2022	10324	5	1106
2	2/1/2022	10145	5	909
3	3/1/2022	10420	5	394
4	4/1/2022	10191	9	740
5	5/1/2022	10190	3	906
6	6/1/2022	10165	4	942
7	7/1/2022	10186	2	854
8	8/1/2022	10259	6	796
9	9/1/2022	10372	1	654
10	10/1/2022	10459	2	577
11	11/1/2022	10410	9	709
12	12/1/2022	10402	1	939
…	...	…	…	…
135	15/5/2022	10686	1	720

Table 2: COVID-19 dataset after normalization

xd_1	xd_2	xd_3	…	xd_16	Target
0.05162	0.05073	0.05210	…	0.00463	0.05337
0.05073	0.05210	0.05096	…	0.00383	0.05432
0.05210	0.05096	0.05095	…	0.00359	0.05271
0.05096	0.05095	0.05083	…	0.00421	0.05326
0.05095	0.05083	0.05093	…	0.00284	0.05273
0.05083	0.05093	0.05130	…	0.00440	0.05262
0.05093	0.05130	0.05186	…	0.00457	0.05294
0.05130	0.05186	0.05230	…	0.00448	0.05474
0.05186	0.05230	0.05205	…	0.00392	0.05624
0.05230	0.05205	0.05201	…	0.00380	0.05703
0.05205	0.05201	0.05263	…	0.00329	0.05840
0.05201	0.05263	0.05302		0.00350	0.06042
…	…	…	…	…	…
0.30348	0.29235	0.26943	…	0.02504	0.12483

3.3. *Comparing the steadiness of both networks*

The network has architecture of 16-5-1, which means that the network consists of three layers; the input layer which consists of 16 neurons, a hidden layer of 5 neurons and an output layer of 1 neuron. Tables 3 and 4 show the performance comparison between Fletcher–Reeves and gradient descent method with various architectures. Furthermore, in the backpropagation neural network (BPNN) algorithm with the gradient descent method, when the number of hidden neurons increases, the means square error (MSE) resulted in an increase for both training data and testing data. It means that an underfitting condition in the backpropagation neural network (BPNN) algorithm with the gradient descent method occurs. Contrastingly, an underfitting condition occurs in both the backpropagation neural network (BPNN) algorithm with the

gradient descent and Fletcher–Reeves method. When number of hidden neurons is larger, then it will lead to underfitting of both networks.

Table 3: Evaluation results using training data for several architectures with a learning rate of 0.005

Architecture	MSE		Computational time	
	Gradient descent	Fletcher–Reeves	Gradient descent	Fletcher–Reeves
16-5-1	0.0400	0.0453	1.0739	0.8462
16-20-1	0.0452	0.0434	1.2457	0.6226
16-50-1	0.0544	0.0458	1.7174	0.6562
16-100-1	0.0529	0.0442	1.4911	1.2539
16-150-1	0.0549	0.0483	1.5547	0.5809

As we increase the number of neurons in the hidden layers of both networks like in Table 4 the change was negative, but the backpropagation neural network (BPNN) algorithm with the Fletcher–Reeves method kept a 2.416 times better results regarding the testing mean square error (MSE) in its worst case, in the 16-5-1 architecture it recorded a 15.68 times less error, the increase of the occurrence of the underfitting conditioning was present in both networks but the slope of negative change was much higher in the backpropagation neural network (BPNN) algorithm with the gradient descent method.

Table 4: Evaluation of the performance of the prediction method for the number of COVID-19 sufferers by using testing data for several architectures for a learning rate equal to 0.005

Architecture	MSE	
	Gradient descent	Fletcher–Reeves
16-5-1	0.0028	0.00026273
16-20-1	0.0035	0.00029107
16-50-1	0.0135	0.00056110
16-100-1	0.0090	0.00310000
16-150-1	0.0084	0.00290000

3.4. *Comparing the impact of changing the learning rate*

Table 5 shows that as the learning rate increased the backpropagation neural network (BPNN) algorithm with the gradient descent method had a steady positive change in both training and testing data, but as we increase the learning rate to 0.1 the error dropped (accuracy increased) steadily, so overall the increase in the learning rate had a positive impact in the backpropagation neural network (BPNN) algorithm with the gradient descent method, on the other side the change in the backpropagation neural network (BPNN) algorithm with the Fletcher–Reeves method was negligible.

In regard to the gradient descent method, the learning rate of 0.2 has resulted in the highest ability to fit the training data. However, the learning rate of 0.1 has resulted in the highest ability to generalize over the testing data, leading to the smallest MSE of the testing data among all the learning rates. The remaining learning rates of 0.01, 0.005 and 0.001 have resulted in a higher MSE for both the training and the testing data. Based on the previous discussion, the learning rate of 0.1 is the best choice for this method since it has the best MSE among the testing data while still having a very small MSE among the training data.

Table 5: Evaluation of the performance of the prediction method for the number of COVID-19 sufferers by using training data for several learning rates with 50 hidden neurons

Learning Rate	MSE train		MSE test	
	Gradient descent	Fletcher–Reeves	Gradient descent	Fletcher–Reeves
0.001	0.0574	0.0435	0.0177	0.0012
0.005	0.0626	0.0434	0.0326	0.0003
0.01	0.0492	0.0474	0.0065	0.0006
0.1	0.0482	0.0439	0.0018	0.0016
0.2	0.0468	0.0477	0.0020	0.0009

In regard to the Fletcher –Reeves method, the learning rate of 0.005 has resulted in the best ability to both fit the training data as well as to generalize over the testing data. The higher and lower learning rates of 0.001, 0.1, 0.01 and 0.2 has resulted in an increase in the mean square error (MSE) for both the training and the testing data. It leads to the conclusion that having a very high or a very low learning rate would not lead to the best training. However, the best learning rate lies on a balanced point between the two.

3.5. *Comparing performance of both networks*

When the training was performed, the data were not split exactly the same –into training and testing - amongst the gradient descent and the Fletcher-Reeves methods. However, similar range of data and similar pattern of division was used. The comparison between the two methods will be performed using the MSE which is dependent on the collective error amongst all the data and does not rely on the comparison between two exactly similar points of data.

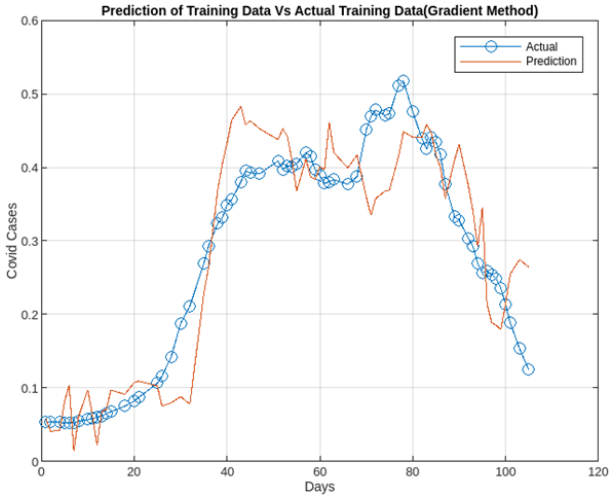

Figure 3: The prediction and actual training data graph by gradient descent method

Figure 3 explain about the training of gradient descent. The backpropagation neural network (BPNN) algorithm with the gradient descent method had an average error on the first 38% of the data, but once the gradient changed the network had a negative performance and even overfitting on the range of Day 65 to Day 85.

Figure 4 explain on training of Fletcher-Reeves method. On the other hand, the backpropagation neural network (BPNN) algorithm with the Fletcher–Reeves method had low error on the complete range of the data, even on the intervals with high rate of change.

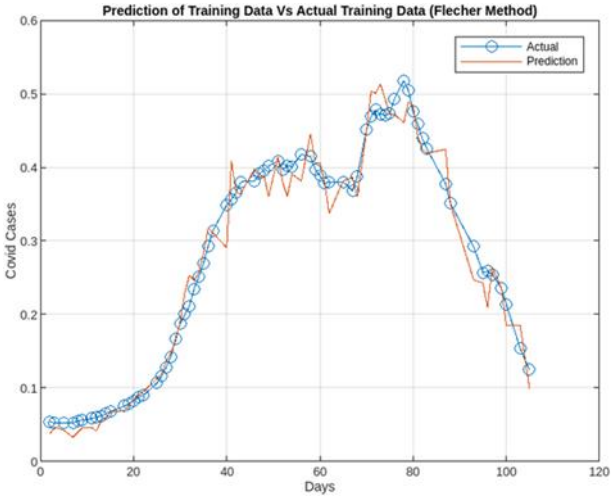

Figure 4: The prediction and actual training data graph by Fletcher-Reeves method

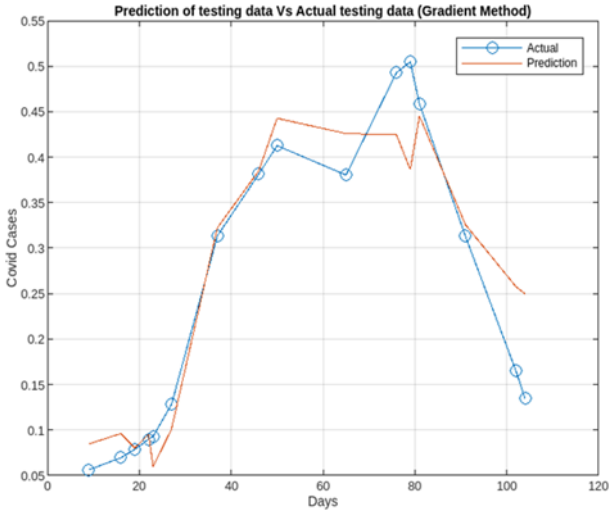

Figure 5: The prediction and actual testing data graph by gradient descent method

Figure 5 discuss on the testing of gradient descent method. The backpropagation neural network (BPNN) algorithm with the gradient descent method had low error on the first 45% of the data, then it started to overshoot the whole another 55%.

Figure 6 discuss on testing of Fletcher-Reeves method. The backpropagation neural network (BPNN) algorithm with the Fletcher–Reeves method had low error on the testing data, except for a small range of error on the oscillating interval Day 63 to Day 87.

According to Table 6, for the BP algorithm with gradient descent method, the learning rate of 0.1 has shown to give the best performance when compared to other learning rates on a constant network architecture of 16-50-1. However, the learning rate of 0.005 has shown the fastest conversion rate with only 173 epochs needed. For the BP algorithm with Fletcher-Reeves method, the learning rate of 0.01 has shown to give the best performance when compared to other learning rates. However, the learning rate of 0.2 has shown the fastest conversion rate with only 10 epochs needed.

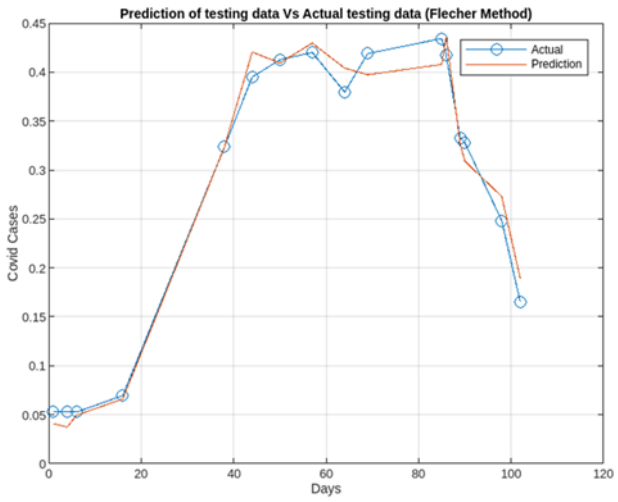

Figure 6: The prediction and actual testing data graph by Fletcher-Reeves method

Table 6: Evaluation results using best performance validation for several learning rates with 50 hidden neurons

	Gradient Descent		**Fletcher-Reeves**	
Learning rate	**Validation Performance**	**Number of Epoch**	**Validation Performance**	**Number of Epoch**
0.005	0.011032	173	0.00060039	33
0.001	0.021713	1000	0.00019702	46
0.01	0.010166	1000	0.00019529	54
0.1	0.00081994	1000	0.0027791	15
0.2	0.00085543	260	0.0018354	10

Table 7: Evaluation results using best performance validation for several architectures with the learning rate of 0.005

	Gradient Descent			**Fletcher-Reeves**		
Architecture	**Validation Performance**	**Number of Epoch**	**Time taken**	**Validation Performance**	**Number of Epoch**	**Time taken**
16-5-1	0.0021296	1000	1.0739	0.00012815	19	0.8462
16-20-1	0.0089344	1000	1.2457	0.00086228	15	0.6226
16-50-1	0.0118950	1000	1.7174	0.00026666	24	0.6562
16-100-1	0.0215570	1000	1.4911	0.00199440	20	1.2539
16-150-1	0.0097314	1000	1.5547	0.00164200	23	0.5809

Based on the Table 7, for the BP algorithm with gradient descent method, the network architecture of 16-5-1 has given the best performance when compared to other architectures on a constant learning rate of 0.005. One more advantage for this architecture is that it has the best computation time of 1.0739 minute among all the other models. For the BP algorithm with Fletcher-Reeves method, the network architecture of 16-5-1 has given the best performance when compared to other architectures on a constant learning rate of 0.005 while the best computation time was achieved by the 16-150-1 architecture. Since this type of networks does not take a significantly high computational time, the 16-5-1 architecture would be a suitable choice for our model.

3.6. *Prediction*

As mentioned before, the data is normalized, so the output of models is not the number of confirmed cases, but it's a value the indicates it, so in order to get the value of confirmed cases we will have to reverse the normalization by multiplying the outputs with $2x10^5$ so we get the value of confirmed cases, which was estimated to be 378 cases.

The estimation was performed using the Fletcher-Reeves method for the 16th May 2022 in Selangor. The actual number of newly confirmed cases was 707 versus an estimation of 378 cases for that day. Based on the previous numbers, the estimation was not highly accurate which is common among prediction models and not highly far away from the actual data. Some of the reasons behind the inaccuracy of the model is the inconsistency of many factors like the social distancing measures, weather temperature and the holiday's timing. Future improvements to the prediction model can be performed through training the model on a larger data set by including a wider range of time as well as more areas in the dataset. Another improvement is to include more factors in the prediction model like the temperature, population density and social distancing measures.

4. Conclusion

4.1. *Conclusion*

The Fletcher–Reeves algorithm has higher overall testing, training, and validation accuracies and it is adaptive to the change in the learning rate, to an extent then the network starts to underfit, it also overfits the data if the network was too complex for the data (as in the example 16-150-1), apart from the two examples given, the Fletcher–Reeves algorithm had a striking performance compared to the gradient descent.

As for the gradient descent, the network mostly converged in much higher time with higher errors compared to the Fletcher–Reeves, but in the network adapted in both the increase of the learning rate and the increase in number of hidden neurons in the hidden layer, overall, the gradient descent is steadily converging to a higher accuracy (slowly) and with no oscillation (overshooting).

As a recommendation, the Fletcher–Reeves had an overall of comprehensive training, the parameters that reflected the highest training-testing accuracies were a learning rate of 0.005 and a network architecture of 16-5-1. The architecture 16-5-1 performed the best among the tested configurations in predicting the target variable. The best model with architecture 16-5-1 and learning rate of 0.005 was trained and evaluated using validation dataset. The training process spanned a total of 19 epochs. After each epoch, the model's performance was measured in terms of accuracy which is 0.00012815. With a learning rate of 0.005, the model achieved an accuracy of 95% on the validation dataset. This indicate that the model effectively learned from the training data and generalized well to unseen data. Additionally, the learning rate of 0.005 ensured a smooth optimization process, preventing overshooting and converging to stable solution.

4.2. *Further study*

For future studies, we could increase the number of hidden layers of the network might have a great positive impact on the network, as deeper networks tend to extract non-shallow features. Another suggestion would be to apply Bayesian Regularization terms on the model as Fletcher-Reeves tend to oscillate as it gets closer to the minimal error, finally applying a searching

algorithm to obtain the neural network's hyper parameters would decrease the computational cost.

4.3. *Limitation of study*

There certain limitations and challenges while using backpropagation with Fletcher-Reeves in predicting analyses of COVID-19 patient, there are some important limitations to consider. Firstly, data quality and quantity; the success of any machine learning model, including neural networks, heavily relies on the quality and quantity of data available for training. In this case of COVID-19 patients, obtaining large and high-quality datasets quite challenging, especially considering the sensitivity of medical data and privacy concerns associated with patient information. Limited or biased data can lead to overfitting or poor generalization of the model. Secondly, model overfitting; neural networks, especially when using complex architecture, can be due to overfitting the training data, meaning that it performs well on the training data but poorly on unseen data. Proper regularization techniques and validation strategies should be employed to resolve this issue.

Furthermore, real-time prediction; in some cases, predictions for COVID-19 patient analyses may be required in real-time to guide immediate medical decisions. The computational complexity and training time of neural networks could be a limitation in such scenarios. Lastly, concept drift; the characteristics of COVID-19 and its patient data may change over time due to evolving variants, treatments protocols, or public health interventions. The model should be periodically reevaluated and retrained to adapt to such concept drift. Addressing these limitations requires careful consideration and multidisciplinary approach that involves medical experts, data scientists and ethicists working collaboratively to develop and validate models that can be responsibly applied in predicting analyses of COVID-19 patients in Selangor.

References

Anam S., Maulana M.H.A.A., Hidayat N., Yanti I., Fitriah Z. & Mahanani D.M. 2021. Predicting the number of COVID-19 sufferers in Malang city using the backpropagation neural network with the Fletcher–Reeves method. *Applied Computational Intelligence and Soft Computing* **2021**: 1–9.

Chong E.K.P. & Żak S.H. 2013. *An Introduction to Optimization*. 4th Ed. Hoboken, New Jersey: John Wiley & Sons.

Eom K., Jung K. & Sirisena H. 2003. Performance improvement of backpropagation algorithm by automatic activation function gain tuning using fuzzy logic. *Neurocomputing* **50**: 439–460.

Fletcher R. & Powell M.J.D. 1963. A rapidly convergent descent method for minimization. *The Computer Journal* **6**(2): 163–168.

Fletcher R. & Reeves C.M. 1964. Function minimization by conjugate gradients. *The Computer Journal* **7**(2): 149–154.

Hestenes M.R. & Stiefel E. 1952. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards* **49**(6): 409–436.

Huang H.-Y. 1970. Unified approach to quadratically convergent algorithms for function minimization. *Journal of Optimization Theory and Applications* **5**(6): 405–423.

Maier H.R. & Dandy G.C. 1998. The effect of internal parameters and geometry on the performance of back-propagation neural networks: an empirical study. *Environmental Modelling & Software* **13**(2): 193–209.

Russell S.J. & Norvig P. 2010. *Artificial Intelligence: A Modern Approach*. 3rd Ed. Upper Saddle River, New Jersey: Pearson Education, Inc.

Thimm G., Moerland P. & Fiesler E. 1996. The interchangeability of learning rate and gain in backpropagation neural networks. *Neural Computation* **8**(2): 451–460.

Department of Mathematics and Statistics
Faculty of Science
Universiti Putra Malaysia
43400 UPM Serdang
Selangor DE, MALAYSIA
E-mail: amirahkhan99@gmail.com, smahani@upm.edu.my[*]

Received: 5 May 2023
Accepted: 6 September 2023

[*]Corresponding author