RESEARCH ARTICLE OPEN ACCESS

Predıctıon Of Breast Cancer Usıng Artıfıcıal Neural Networks

Mariya Kiknadze*,Ahmet Gürhanlı**
**Istanbul Aydın Üniversity, Computer Engineering Department, Istanbul*
***İstanbul Aydın Üniversity, Computer Engineering Department, İstanbul*
Corresponding author: Mariya Kiknadze

ABSTRACT
Breast cancer is one of the most important malignant diseases in the world. In the United States, breast cancer ranks first among all oncological diseases in women and is the second leading cause of cancer mortality after lung cancer. Despite recent great success in the early detection and treatment of breast cancer, new approaches and algorithms are still being developed for early diagnosis. Breast cancer has many classifications, like other malignant diseases: histological, molecular, functional, TNM classification. Most cases of cancer can be diagnosed in the later stages of the disease, and treatment is often not responding and the patient is lost. Therefore, early detection of breast cancer is vital. This study uses the UCI Breast Cancer Wisconsin (Diagnostic) Data Set (WDBC), which is presented by measuring test classification accuracy, sensitivity, and specificity values. The data set was divided into 70% for the training phase and 30% for the testing phase. This study demonstrates the importance of optimization algorithm selection and parameters in the diagnosis of Breast Cancer using Artificial Neural Networks and investigates how they should be chosen. The accuracy results of different optimization algorithms and parameter values are reported.

Keywords: Artificial Neural Networks; Breast Cancer;Breast Cancer Diagnosis

Date of Submission: 28-01-2020 Date Of Acceptance: 11-02-2020

I. INTRODUCTION

Machine learning is currently used in many fields of science and production. Medicine is no exception to this field. Thanks to machine learning, many tasks such as classifying patients, determining the most appropriate treatment method, estimating the duration and outcome of a disease, evaluating the risk of complications, finding the most characteristic syndromes of a particular disease type are solved. Breast cancer is a disease caused by normal glandular cells turning into cancer. In the world, breast cancer is the most common form of cancer among women. Among women, between 13 and 90 years old, one in 13 or one in 9 is infected with this disease [1]. As with many other cancers, early diagnosis of breast cancer can save patients' life. Therefore, making a definitive diagnosis of breast cancer in the early stages is very important to keep the patient's quality of life at the best possible level. However, even regular mammograms do not guarantee a timely diagnosis of this disease. US scientists have developed data-based software to automatically classify breast density and thus detect breast cancer. Tests have shown that this system is as accurate as a "diagnostician" as human radiologists [2]. This algorithm can help doctors in cases where the density of the chest does not allow a clear diagnosis. The application of machine learning models has become an integral part of cancer studies, which later aims to improve patients' treatment for disease prediction and prognosis. Two sets of data from relevant breast cancer studies are combined using a data integration approach based on horizontal and vertical integration using appropriate data with good performance and no data loss and graphical databases.

It is known that Donald Hebb (1949) found the theory of modern neural networks. The neurologist Hebb studied how the brain learned. The basic unit of the work of the brain is how the nerve cell relates to the two nerve cells, and is based the theory of neural networks on this basis. Based on this foundation of Hebb, the idea was launched and has hundreds of theories. data set in breast cancer, it is critical to apply Today, there are many an artificial neural network models with a 99% success rate used in our real life. Machine learning with artificial neural network, image processing [3] [4], character definition, classification, prediction, clustering, sound processing [5], data filtering and It is possible to make many applications such as getting into the most suitable form.One of the main reasons for using artificial neural networks in these areas is that any kind of data, regardless of the algorithm used, can minimize learning errors and therefore make realistic predictions. In order to make an accurate prediction on the basis of clinicalthe Artificial Neural Network model with the correct optimization algorithm and to determine the parameter ranges correctly.Therefore, our study investigated how the following optimization algorithms worked.

1.1 Stochastic Gradient Descent (SGD)

SGD deep learning, the objective function is usually the average value of the loss functions for each sample in the training data set. We assume that $F_i(x)$ n data, index i and parameter vector x and training data example are a loss function, after that we have an objective function.

$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i\ (x)$ (1)

The gradient of the objective function in x is calculated as follows:

$\nabla f(x) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x)$ (2)

The calculation cost for each iteration of the independent variable is O (n), which grows linearly with n, if gradient descent is used. Therefore, the gradient landing cost for each iteration will be very high when model’s training data sample is large.

The calculation reduces cost of each iteration from stochastic gradient descent. For each iteration of the stochastic gradient descent, we select the index i∈ {1, ..., n} for random data samples and calculate the gradient $\nabla f_i(x)$ to update x:

$x \leftarrow x - \eta\nabla f_i(x)$ (3)

Here learning rate is η. We can see that the cost of calculation for each iteration falls from the gradient descent of O (n) to constant O (1). It should be noted that the stochastic gradient $\nabla f_i(x)$ is an unbiased estimate of the gradient ∇f (x) [6].

$E_i\nabla f_i(x) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x) = \nabla f(x)$ (4)

This means, on average, that the gradient of the stochastic gradient is a good estimate.

1.2 Adaptive Gradient Descent (Adagrad)

We use the s_t variable to accumulate the past gradient variance as follows.

$$g_t = \partial_\omega l\ (y_t, f(x_t, \omega)),$$

$s_t = s_{t-1} + g_t^2,$

$\omega_t = \omega_{t-1} - \frac{\eta}{\sqrt{s_t+\epsilon}} \cdot g_t.$ (5)

Learning rate is η and ϵ is a contribution constant that allows us to divide by 0. Finally, we start the value of $s_0 = 0$.

As with momentum, we need to follow the auxiliary variable in this case to take into account the individual learning speed for each coordinate. This does not significantly increase Adagrad's cost compared to SGD, since the main cost usually consists of calculating l $(y_t, f(x_t, \omega))$, and its derivative. As with momentum, we need to track an auxiliary variable, in this case Adagrad's cost relative to SGD since Adagrad calculates l $(y_t, f(x_t, \omega))$, and derivative to allow an individual learning rate per coordinate does not significantly increase.

1.3 Root Mean Square Propagation (RMSprop)

The RMSprop algorithm is used as a simple fix that allows separating speed planning from coordinate adaptive learning speeds. The problem is that Adagrad accumulates the squares of the g_t gradient in the status vector$g_t = s_t - 1 + g_t^2$. As a result, because the algorithm converges, $s_t$continues to grow, mainly without linear restrictions, due to a lack of normalization (Ruder 2017) [6].One way to solve this problem would be to use s_t / t. For reasonable g_t deployments, this will converge. Unfortunately, since the procedure remembers the full trajectory of values, it may take a very long time for the behavior of the limit to become important. An alternative gives RMSprop to keep the average leak value for some parameters $s_t \leftarrow \gamma s_{t-1} + (1-\gamma)g_t^2 \gamma > 0$ and all other parts unchanged.

$$s_t \leftarrow \gamma s_{t-1} + (1-\gamma)g_t^2$$

$x_t \leftarrow x_{t-1} - \frac{\eta}{\sqrt{s_t+\epsilon}} \odot g_t$ (6)

The constant ϵ> 0 is usually set to 10^{-6} to ensure that we are not divided into zero or very large pitch sizes. Given this expansion, η controls the speed of learning independently of the scaling applied per coordinate.

1.4 Adaptive Learning Rate (Adadelta)

In Adadelta is another version of AdaGrad. The difference is that the speed of learning reduces the amount of adaptation to coordinates. It also uses the amount of change as a calibration for future change, since it doesn't traditionally have a learning rate. In summary, Adadelta uses two state variables: s_t to store the average leak of the second moment of the gradient, and Δx_t to store the average leak of the second change moment in the model itself.

$s_t = ps_{t-1} + (1-p)g_t^2,$

$\mathbf{g'_t = \sqrt{\frac{\Delta x_{t-1}+\epsilon}{s_t+\epsilon}} \odot g_t,}$

$\mathbf{x_t = x_{t-1} - g'_t,}$

$\Delta x_t = p\Delta x_{t-1} + (1-p)x_t^2.$ (7)

The difference from the previous one is that we perform updates with a modified gradient g'_t, which is calculated by taking the relationship between the average square of the rate of change and the average second moment of the gradient. The use of $g'_t$is for ease of identification only. In practice, we can implement this algorithm for g'_t without having to use additional temporary space. As before, η is a parameter that provides non-trivial numerical results, that is, avoids zero-step magnitude or infinite variance general, it is set to $\eta = 10^{-5}$.

1.5 Adaptive Moment Estimation (Adam)

One of Adam's main components is that he uses exponentially leaky averages to get an estimate of both the momentum and the second moment of the gradient. So, it uses state variables

$$v_t \leftarrow \beta_1 v_{t-1} + (1-\beta_2)g_t^2$$
$$s_t \leftarrow \beta_2 s_{t-1} + (1-\beta_2)g_t^2 \quad (8)$$

Here β1 and β2 are non-negative weight parameters. The normal choice for them is: $\beta_1 = 0.9$ and $\beta_2 = 0.999$.. That is, the variance moves much slower than the momentum term. If we start the value of $v_0 = s_0 = 0$, we initially have a significant bias to the lower values. This can be resolved using $\sum_{i=0}^{t}\beta^i = \frac{1-\beta^t}{1-\beta}$ to normalize terms. Accordingly, normalized state variables

$$\hat{v}_t = \frac{v_t}{1-\beta_1^t} \text{ and } \hat{s}_t = \frac{s_t}{1-\beta_2^t} \quad (9)$$

Now we can write the update equations and first, they are scaled very similar to RMSProp to get the gradient.

$$g'_t = \frac{\eta\widehat{v_t}}{\sqrt{\widehat{s_t}+\epsilon}}. \quad (10)$$

Unlike RMSprop, our update uses the momentum $\widehat{v_t}$ instead of the gradient itself. Moreover, there is a small cosmetic difference, since rescaling takes place using $\frac{1}{\sqrt{\widehat{s_t}+\epsilon}}$ instead of $\frac{1}{\sqrt{\widehat{s_t}+\epsilon}}$. It works a little better in practice than in the previous practice, so the deviation from RMSProp usually we choose $\epsilon = 10^{-6}$ for a good balance between numerical stability and fidelity. We have all the parts to calculate the updates, this is a bit anti-eclectic and we have a simple update of the form. Then they use it to update parameters as we see in Adadelta and RMSprop. The man gives the update rule:

$$\omega_{t+1} = \omega_t - \frac{\eta\widehat{v_t}}{\sqrt{\widehat{s_t}+\epsilon}} \quad (11)$$

1.6 Adaptive Moment Estimation Maximum (AdaMax)

The s_t factor in the man update rule scales the gradient inversely proportional to the norm l_2 and is the past gradients (via the term v_{t-1}) and the current gradient is $|g_t|$ (Ruder 2017)[6].

$$s_t \leftarrow \beta_1 s_{t-1} + (1-\beta_2)|g_t|^2 \quad (12)$$

By adapting this to the Adam update, we get the AdaMax update rule.

$$\omega_{t+1} = \omega_t - \frac{\eta}{s_t}\widehat{v_t} \quad (13)$$

Her

$$S_t = \beta_2^{\infty} s_{t-1} + (1-\beta_2^{\infty})|g_t|^{\infty} = max(\beta_2 s_{t-1,}|g_t|) \quad (14)$$

1.7 Nesterov-accelerated Adaptive Moment Estimation (Nadam)

Nesterov accelerated Adaptive Moment Estimation combines Adam and NAG. To include Nesterov accelerated gradient (NAG) in Adam, we need to change the term momentum [6]

$$g_t = \nabla_{\omega_t} J(\omega_t)$$
$$m_t = \gamma m_{t-1} + \eta g_t$$
$$\omega_{t+1} = \omega_t - m_t \quad (15)$$

Here our objective function is J, momentum distortion term γ and η are step size. The expansion of the third equation above gives:

$$\omega_{t+1} = \omega_t - (\gamma m_{t-1} + \eta g_t) \quad (16)$$

It shows once again that it contains a step in the direction of the momentum vector and the current gradient. Nesterov updates the accelerated gradient, allowing us to take a more precise step in the direction of the gradient. So, by simply applying the forward momentum vector to update the current parameters to find the NAG value in g_t:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)g_t$$
$$\widehat{m_t} = \frac{m_t}{1-\beta_1^t}$$
$$\omega_{t+1} = \omega_t - \frac{\eta\widehat{m_t}}{\sqrt{\widehat{s_t}+\epsilon}} \quad (17)$$

$\widehat{m_{t-1}}$ momentum vector gives the bias correction estimate of current $\widehat{m_t}$ momentum vector Nadam update rule.

$$\omega_{t+1} = \omega_t - \frac{\eta}{\sqrt{\widehat{s_t}+\epsilon}}\left(\beta_1\widehat{m_t} + \frac{(1-\beta_1)g_t}{1-\beta_1^t}\right) \quad (18)$$

II. LITERATURE RESEARCH

Many studies have been done on this subject. In fact, where machine learning combines mathematical statistics, optimization methods and classical mathematical disciplines also have their own characteristics associated with computational efficiency and retraining problems. Many inductive training methods have been developed as an alternative to classical statistical approaches. Many methods in studies are closely related to the extraction of information and artificial data analysis.

Xrulyov K.A.ve Ryazanov M.A. (2016) developed a web service with Azure Machine Learning using the analysis of data about patients examined for diagnosis of breast cancer.

Fogel D. B., Wasson E.C., Boughton E.M. and Porto V.W. (1997) conducted data analysis studies for the detection of breast cancer using neural networks with patient age using radioactive features [7].

Revett K., Gorunescu F., Gorunescu M., El-Darzı E. and Ene M. (2005) and Gorunescu M., Gorunescu F., and Revett K. (2007) hybrid with raw clusters and possible neural networks They developed a decision support system for a breast cancer medical model based on a model[8].

[9]. Hsiao Y.H., Huang Y.L., Liang W.M., Kuo S.J. and Chen D.R. (2009) conducted an MLP classifier analysis study to identify benign or malignant breast tumors using vascular parameters (harmonic and non-harmonic 3D Dopplerography).

E.Harwich, K.Laycock., (2018) and JASON The MITER Corporation (2017) British scientists in "UK" "Artificial Intelligence in the National Health System" and USA's leading American technology scientist Jason "Health and Health Services For Artificial Intelligence ", they published a report in 2017. In both studies, providing high-quality medical care to the general population by using Artificial Intelligence was analyzed. In the field of cancer diagnosis, a study was conducted on the use of Artificial Intelligence, the functions and methods of Artificial Intelligence[10] [11].

Mihaylov.I, Nisheva.M, and Vassilev.D (2019) conducted a study to predict the survival time in breast cancer based on clinical data using machine learning models for an accurate diagnosis. In this study, he estimates that the patient's survival time, tumor stage, tumor size, and age are clinical features integrated with the originally developed tumor. In addition to data normalization and classification in the study, the applied machine learning method gives promising results in terms of the accuracy of the survival estimate [12]. In this study, linear support vector regression, core ridge regression, K nearest neighbor regression, decision tree regression, and lasso regression models obtained the most accurate life prognosis results. Using the same methods, they developed a Python-based workflow as the proposed approach to performance on breast cancer data.

III. DATA SET USED IN BREAST CANCER ESTIMATION

The data set used to diagnose breast cancer using the Artificial Neural Network is the breast cancer (BC) database from the University of California (UCI) Machine Learning Depot [2]. An open repository with many datasets that can be used for experimental analysis of machine learning algorithms from the University of Wisconsin Hospital in Madison. A data set created by William H. Wolberg. Some of the features in the dataset used in this study are radius, texture, perimeter, softness, compactness, area, concaveness, concave points, fractal size for each cell nucleus, symmetry and a Wisconsin Diagnostic consisting of 329 functions and 32 functions for machine learning. Breast Cancer (WDBC) data set. 458 of 699 breast cancer data in the WDBC dataset show examples of benign and 241 of malignant cancer cells. Distribution of benign cancer cells is more homogeneous in the dataset, and malignant cancer cells have structural malignancies. A total of 11 attributes and value ranges in the database are shown in the table below.

Table 1: Data Sets and value ranges

Data sets	Value ranges
Sample code number	ID Numarası
Clump Thickness	1-10
Uniformity of Cell Size	1-10
Uniformity of Cell Shape	1-10
Marginal Adhesion	1-10
Single Epithelial Cell Size	1-10
Bare Nuclei	1-10
Bland Chromatin	1-10
Normal Nucleoli	1-10
Mitoses	1-10
Class	2 / 4

Ten of the attributes used in the estimation of breast cancer diagnosis application after the id number are 1 to 10 attributes. Our last value is seen as the result part, and if it is benign, it takes the value of 2, if it is malignant, it takes the value of 4. The first ten data of the data set to be used are given in Table 2.

Table 2: Top Ten Data in the Data Set

Sample code number	Clump Thickness	Uniformity of Cell Size	Uniformity of Cell Shape	Marginal Adhesion	Single Epithelial Cell Size	Bare Nuclei	Bland Chromatin	Normal Nucleoli	Mitoses	Class
1000025	5	1	1	1	2	1	3	1	1	2
1002945	5	4	4	5	7	10	3	2	1	2
1015425	3	1	1	1	2	2	3	1	1	2
10162	6	8	8	1	3	4	3	7	1	2

77									
1017023	4	1	1	3	2	1	3	1	1	2
1017122	8	10	10	8	7	10	9	7	1	4
1018099	1	1	1	1	2	10	3	1	1	2
1018561	2	1	2	1	2	1	3	1	1	2
1033078	2	1	1	1	2	1	1	1	5	2
1035283	1	1	1	1	1	1	3	1	1	2

Since the id number of the data in the data set to be used will not cause any change to the result, we subtract this field, and since our artificial neural network model has two outputs in the part of whether it is benign or malignant, we need to add two fields and indicate this situation.
In the last case, the format in which the ten data in table 2 will be understood by the application is as in table 3.

Table 3: Top Ten Data of the Reconstructed Data Set for ANN Model

5	1	1	1	2	1	3	1	1	1
5	4	4	5	7	10	3	2	1	1
3	1	1	1	2	2	3	1	1	1
6	8	8	1	3	4	3	7	1	1
4	1	1	3	2	1	3	1	1	1
8	10	10	8	7	10	9	7	1	0
1	1	1	1	2	10	3	1	1	1
2	1	2	1	2	1	3	1	1	1
2	1	1	1	2	1	1	1	5	1
1	1	1	1	1	1	3	1	1	1

IV. USED ARTIFICIAL NEURAL NETWORKS MODEL

In this study, there are 9 attributes in the dataset to diagnose the disease. For this reason, there are 9 neurons in the input layer, and since the tumors are benign or malignant, there are 2 neurons in the output layer and 10 neurons in the intermediate layer. At the beginning of the application, random bias and weight values were created and the last values were found by updating these values during the application period. In practice, the probability of error in the evaluation is minimized due to the spread of the error in the output neurons.
As the activation function, the sigmoid function was preferred because it is easy to distinguish.

$$\boldsymbol{sigmoid(s)} = \frac{\mathbf{1}}{\mathbf{1+e^x}} \qquad (19)$$

Calculations were made according to the formulas below.

$$\boldsymbol{\theta_n = f\left(\sum_{i=1}^{9} x_i.\,\omega_{in} + b_n\right), n = 1, \ldots, 10.} \qquad (20)$$

Here, b_n and in mean the bias and weights of the intermediate layer, θ_n means the output of the intermediate layer.

$$\boldsymbol{Y_n = f\left(\sum_{i=1}^{10} \theta_i.\,\omega'_{in} + b'_n\right), n = 1, 2.} \qquad (21)$$

Here, Y_n represents the output of the last layer. b_n and ω_{in} are the output layer and their weights. Here the learning rate is taken as 0.5.

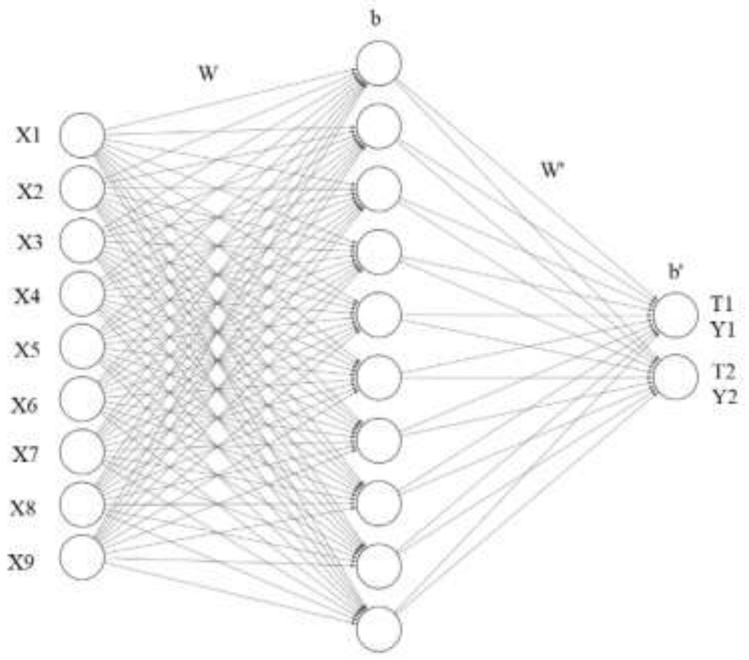

Figure 1. Artificial Neural Network Model in the Study

Updates were made in the study with the following formulas.

$$\boldsymbol{\delta_i = \left(T_J - Y_j\right).Y_{J.}\left(1 - Y_j\right)}$$

$$\boldsymbol{\Delta W'_{ij} = (t+1) = 0.5.\delta_j.\theta_{i,j} = 1,2 \; ve \; i = 1, \dots, 10.}$$

$$\boldsymbol{W'^{yeni}_{ij} = W'^{eski}_{ij} + \Delta W'_{ij}(t+1)}$$

$$\mathbf{\Delta W_{ij}(t+1) = 0,5.\theta_j.\left(1-\theta_j\right).\delta_{1,2}.W_{ij}.X_i, \; j = 1, \dots, 10 \; ve \; i = 1, \dots, 9.}$$

$$\mathbf{W^{yeni}_{ij} = W^{eski}_{ij} + \Delta W_{ij}(t+1)}(22)$$

$$\mathbf{\Delta b'_{ij}(t+1) = 0,5.\delta_j, i = 1,2.}$$

$$\mathbf{b'^{yeni}_{ij} = b'^{eski}_{ij} + \Delta b'_{ij}(t+1)}$$

$$\mathbf{\Delta b_{ij}(t+1) = 0,5.\theta_{j.}\left(1-\theta_j\right).\delta_{1,2}.W_{j,i}, i = 1, \dots, 10 \; ve \; j = 1, \dots, 9.}$$

$$\mathbf{b^{yeni}_{ij} = b^{eski}_{ij} + \Delta b_{ij}(t+1)}$$

$$\mathbf{hata = \frac{1}{2}\sum_{i=1}^{2}(t_n - y_n)^2}$$

The output values on the target represent $t_{1,2}$ in the formula.

V. SELECTION OF OPTIMIZATION ALGORITHM USED IN ARTIFICIAL NEURAL NETWORK AND ADJUSTMENT OF PARAMETERS

5.1Optimization Algorithm

As an optimization method 'SGD', 'RMSprop' , 'Adagrad' , 'Adadelta', 'Adam' , 'Adamax' and 'Nadam' tests were carried out with all of them to determine the best accuracy rate algorithms [13]. Table 4 lists the formulas used by these algorithms.

Table 4: Formulas of Optimization Functions

Optimization Function	Formulas
Stochastic Gradient Descent (SGD)	$Q(w) = \frac{1}{n}\sum_{i=1}^{n} Q_i(w),$
Root Mean Square Propagation **(RMSprop)**	$w := w - \frac{\eta}{\sqrt{v(w,t)}}\nabla Q_i(w)$
Adaptive Gradient Descent **(Adagrad)**	$G = \sum_{\tau=1}^{t} g_\tau g_\tau^{\mathsf{T}}$

Adaptive Learning Rate **(Adadelta)**	$\Delta x_t = -\frac{\eta}{\mathrm{RMS}[g]_t} g_t$
Adaptive Moment Estimation **(Adam)**	$w^{(t+1)} \leftarrow w^{(t)} - \eta \frac{\hat{m}_w}{\sqrt{\hat{v}_w} + \epsilon}$
Adaptive Moment Estimation Maximum **Adamax**	$w_{t+1} = w_t - \frac{\alpha}{S_t} \cdot \hat{V}_t$
Nesterov-accelerated Adaptive Moment Estimation **(Nadam)**	$w_{t+1} = w_t - \frac{\alpha}{\sqrt{\hat{S}_t} + \epsilon}\left(\beta_1 \hat{V}_t + \frac{1-\beta_1}{1-\beta_1^t} \cdot \frac{\partial L}{\partial w_t}\right)$

Table 5: Average accuracy rate for different optimization methods

Optimizer	**Average accuracy rate**
SGD	0.607
RMSprop	0.957
Adagrad	0.942
Adadelta	0.957
Adam	0.957
Adamax	0.957
Nadam	0.957

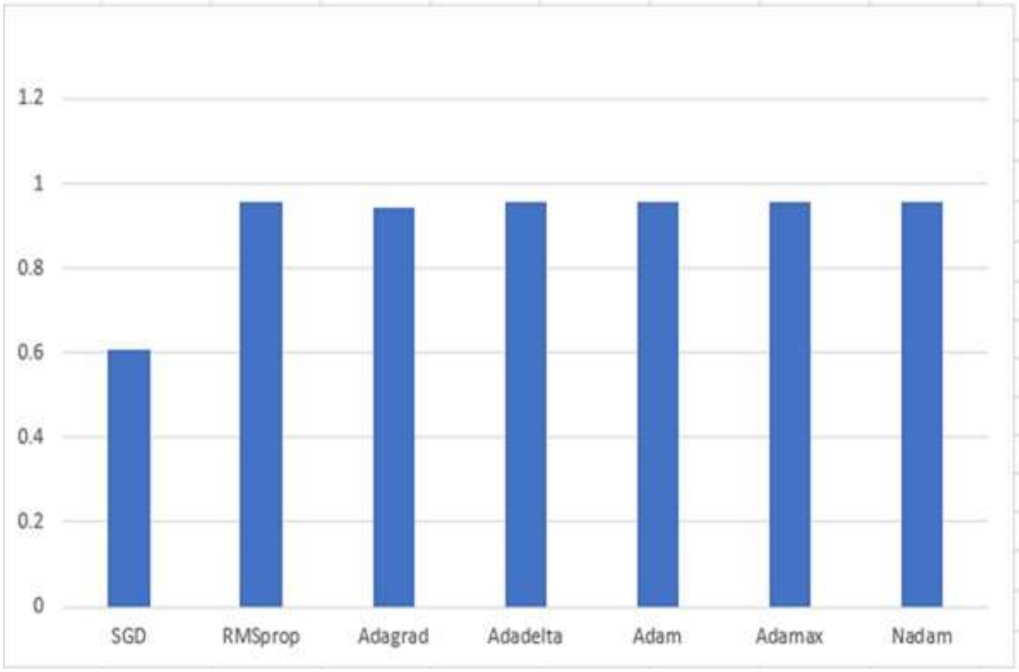

Figure 2. Average accuracy graph for different optimization methods

As seen in the chart in Table 5 and Figure 2, those outside SGD and Adagrad give close accuracy to each other and can be preferred in this application. Although ADAM gives the best accuracy although they give close accuracy values to each other, the next examinations were made using the ADAM algorithm.

5.2 Batch Size

Batch Size determines the number of samples to be distributed over the network. For example, suppose you have 1050 training examples and you want to set the batch size value to 100. The algorithm takes the first 100 samples (1 to 100) from the training dataset and trains the network. Then he takes the second 100 samples (101 to 200) and retrains the network. We can continue doing this until we distribute all the samples over the network. A problem may occur in the last set of samples. In our example, we used 1050, which is not divided by the remaining 100. The simplest solution is to train the network by taking the last 50 samples.

The benefits of using samples by dividing them into groups are as follows. The general training procedure requires less memory as you train the network using fewer samples. This is particularly important where the entire data set does not fit in the machine's memory. Networks often learn faster with mini packets. This is because we update the weight after each spread. In our example, we distributed 11 packages (10 of them were 100 samples and 1 of them were 50 samples) and after each of them, we updated the parameters of our network. If we used all the samples during distribution, only 1 update would be made for the network parameter.

Table 6: Average accuracy rate for Batch Size

Batch Size	Average accuracy rate
10	0.957
20	0.950
40	0.950
60	0.950
80	0.950
100	0.950

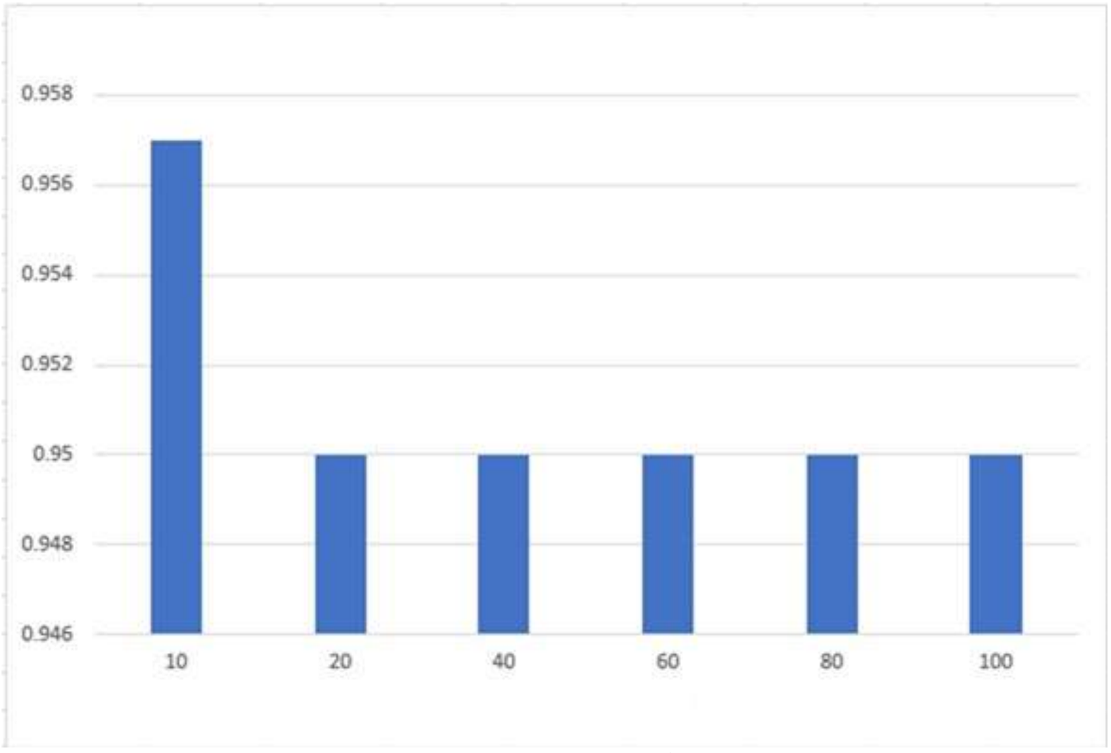

Figure 3. Average accuracy graph for Batch Size

According to the chart in Table 6 and Figure 3, the best result is when the batch is 10.

5.3 Epoch

Epoch is a hyperparameter that determines how many times the learning algorithm runs on the entire set of training data. An epoch means that each instance in the training data set has the opportunity to update the internal parameters of the model. Epoch consists of one or more batches. For example, as mentioned above, a period in which there is a group is called a batch gradient descent learning algorithm. We can consider a for loop for the number of cycles where each cycle goes through a series of training data. For this loop, there is another nested for loop repeated on each sample group, where a group has a certain number of "batch size" samples. The speed is traditionally large usually, hundreds or thousands, which allows the learning algorithm to work until the model error is minimized. In the literature, there are examples of the number of cycles as 10, 100, 500, 1000 and larger. Typically, line graphs are created along the X-axis that represents phases as time and error or model skill on the Y-axis, and these graphs are sometimes called learning curves. These charts can help diagnose whether the model is retrained, not well understood, or suitable for a training dataset.

Table 7: Average accuracy rate for Epoch

Epoch	Average accuracy rate
10	0.607
50	0.950
100	0.950
200	0.957
500	0.964
1000	0.950

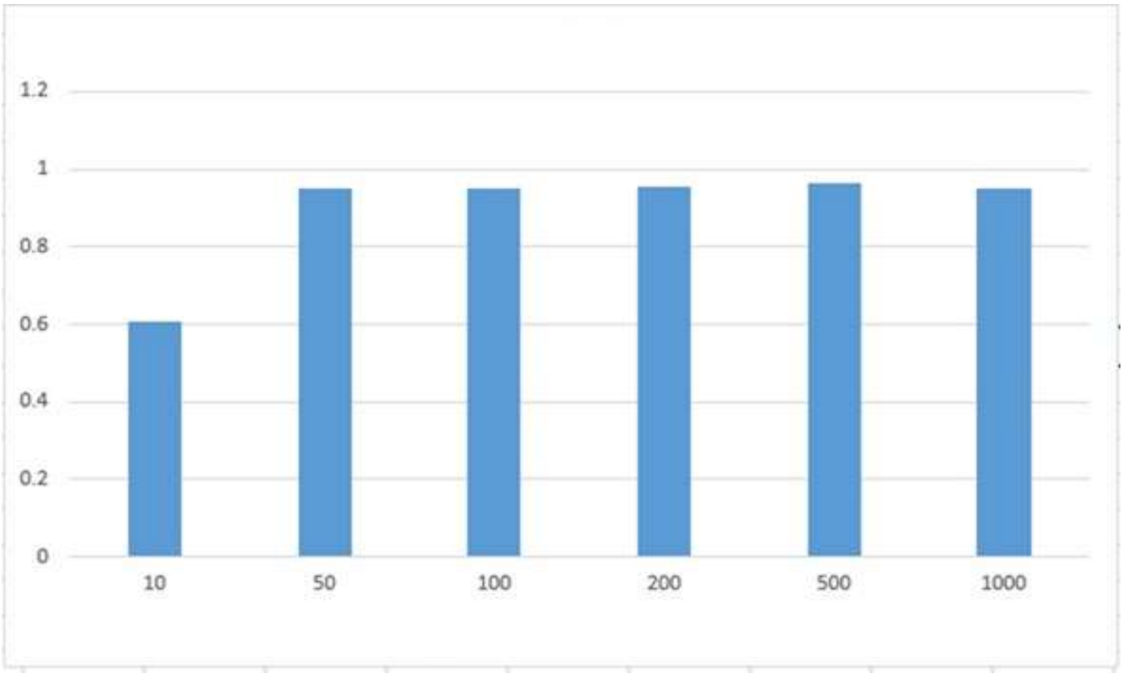

Figure 4. Average accuracy rate graph for Epoch.

As seen in Table 7 and Figure 4, when the epoch is 500 the accuracy rate is the best.

VI. CONCLUSION

Since artificial neural networks do not have a linear structure, it is more important to determine the variables we mentioned in this article correctly. These parameters cannot be given a clear and fixed value, but they also vary according to the type of data set used in the training of the artificial neural network. In our studies, we have seen how the optimization algorithm and parameter values affect the result. When we updated our initial parameters, the average accuracy rate increased from 0.607 to 0.96. The optimization algorithm preference and parameter settings for breast cancer prediction using artificial neural networks with the features in the WDBC dataset gave the best accuracy rate in our tests.

- Optimization Algorithm: ADAM
- Batch Size: 10
- Epochs: 500

REFERENCES

[1]. Aleksandroviç X, K., Ryazanov M.A., (Barnaul2016) http://elibrary.asu.ru/xmlui/bitstream/handle/asu/2682/vkr.pdf?sequence=1&isAllowed=y

[2]. William H Wolberg, W Nick Street, and Olvi L Mangasarian. (1992). Breast cancer Wisconsin (diagnostic) data set. UCI Machine Learning Repository [http://archive. İCS. UCİ. edu/ml/] (1992)

[3]. Shi Z. ve He L.,(2010) "Application of Neural Networks in Medical Image Processing", Proceedings of the Second International Symposium on Networking and Network Security (ISNNS '10) , China, (2010), p. 2-4.

[4]. Ramirez-Quintana J.A., Chacon-Murguia M. I. ve Chacon-Hinojos J. F.,(2012) "Artificial Neural Image Processing Applications: A Survey", Engineering Letters, (2012), 20 (1), p 68-81.

[5]. Uncini A., (2003) "Audio signal processing by Neural Networks", Neurocomputing, (2003), 55 (3-4), p. 593 – 625.

[6]. Sebastian Ruder (2017) "An overview of gradient descent optimization algorithms" Insight Centre for Data Analytics, NUI Galway Aylien Ltd., Dublinarxiv:1609.04747v2 [csLG] 15 June 2017

[7]. Fogel D. B., Wasson E.C., Boughton E.M. ve Porto V.W.,(1997) "A step toward computer-assisted mammography using

evolutionary programming and neural Networks", Cancer Letters, (1997), 119 (1), Sayfa 93-97.

[8]. Revett K., Gorunescu F., Gorunescu M., El-Darzı E. ve Ene M.,(2005) "A breast cancer diagnosis system: a combined approach using rough sets and probabilistic neural Networks", Computer as a tool Eurocon 2005, Belgrade, (2005), pp. 1124- 1127.

[9]. Hsiao Y.H., Huang Y.L., Liang W.M., Kuo S.J. and Chen D.R., (2009) "Characterization of benign, and malignant solid breast masses: harmonic versus nonharmonic 3D power Doppler imaging", Ultrasound Medicine & Biology, (2009), 35 (3), pp. 353-359.

[10]. E.Harwich, K.Laycock., (2018) Thinking on its own: AI in the NHS [Electronic resource] URL: http://www.reform.uk/publication/thinking-on-its-own-ai-in-the-nhs/ (application date17.05.2018).

[11]. JASON The MITRE Corporation (2017) Artificial Intelligence for Health and Health Care [Electronic resource]. URL: https://www.healthit.gov/sites/default/files/jsr-17-task-002_aiforhealthandhealthcare 12122017.pdf (application date17.05.2018).

[12]. Iliyan Mihaylov, Maria Nisheva, and Dimitar Vassilev (2019) "Application of Machine Learning Models for Survival Prognosis in Breast Cancer Studies" Published: 3 March 2019 file:///C:/Users/Admin/Downloads/information-10-00093.pdf

[13]. J. Duchi, E. Hazan, and Y. Singer, (2011) "Adaptive subgradient methods for online learning and stochastic optimization," Journal of Machine Learning Research, vol. 12, no. Jul, pp. 2121–2159, 2011.

[14]. David H. Staelin and Carl H. Staelin (2011) "Models for Neural Spike Computation and Cognition"ISBN 9781466472228 CreateSpace, Seattle, Washington. November 2011.

[15]. G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, (2016) "Deep networks with stochastic depth," in European Conference on Computer Vision. Springer, 2016, pp. 646–661.

[16]. Gorunescu M., Gorunescu F., ve Revett K.,(2007) "Investigating a Breast Cancer Dataset Using a Combined Approach: Probabilistic Neural Networks and Rough Sets", Proc. 3rd ACM International Conference on Intelligent Computing and Information Systems -ICICIS07, Cairo, Egypt, (2007), pp. 246-249.

[17]. M. Ishii and A. Sato, (2017) "Layer-wise weight decay for deep neural networks," in Pacific-Rim Symposium on Image and Video Technology. Springer, 2017, pp. 276–289.

[18]. S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le, (2018) "Don't decay the learning rate, increase the batch size," in International Conference on Learning Representations (ICLR), 2018.

[19]. Y. Nesterov, (1983) "A method for unconstrained convex minimization problem with the rate of convergence o (1/kˆ 2)," in Doklady AN USSR, vol. 269, 1983, pp. 543–547.

[20]. Z. Huo and H. Huang, (2017) "Asynchronous mini-batch gradient descent with variance reduction for non-convex optimization," in Thirty-First AAAI Conference on Artificial Intelligence, 2017.