Chapter 4

Enhancing Diagnostic Precision in Breast Cancer Histopathology: A Novel Weight Optimization Approach for Artificial Neural Network

4.1 Introduction

In the landscape of machine learning, traditional optimization algorithms are not just tools; they are the navigators guiding models through the intricate terrain of parameter space towards optimal performance. These algorithms meticulously calibrate model parameters, minimizing or maximizing a specified objective functionoften a loss function that quantifies the gap between a model's predictions and the actual observations. This process is fundamental to machine learning, as the essence of these models lies in their ability to learn from data and make accurate predictions. Traditional optimization algorithms operate by systematically exploring the model's parameter space. This exploration is a quest to discover the set of parameters that yields the best performance of the model as defined by the objective function. The journey is intricate, involving the assessment of how variations in parameters impact the objective function, and making informed adjustments to steer the model towards improvement. Among the various optimization techniques, gradient-based methods stand out for their widespread application in machine learning. These methods hinge on the concept of gradients, which indicate the direction of the steepest ascent in the function's value. By moving in the opposite direction, these algorithms aim to find the function's minimum valuea strategy that is at the heart of the Gradient Descent algorithm and its numerous variants. Each variant, be it Stochastic Gradient Descent, Mini-batch Gradient Descent, or others like Momentum, AdaGrad, RMSProp, and Adam, is crafted to tackle specific challenges, such as computational efficiency and the erratic nature of real-world data [56]. The challenges that traditional optimization algorithms confront in machine learning are manifold, particularly due to the high-dimensional parameter spaces of complex models like deep neural networks. These high-dimensional spaces are akin to vast, uncharted territories filled with obstacles such as local minima and saddle points, especially prevalent in non-convex optimization problems that are a staple in machine learning. Navigating through this complex landscape requires algorithms that are not just efficient but also astute, capable of discerning the right path towards optimization amidst a myriad of potential missteps [56]. An integral component of these algorithms is their ability to incorporate mechanisms like regularization, a technique that enhances the model's ability to generalize and perform well on unseen data. Regularization modifies the objective function to impose a penalty on complexity, nudging the model towards simplicity and robustness. Alongside regularization, traditional optimization algorithms adeptly handle constraints, ensuring that the optimized parameters remain within predefined bounds or satisfy certain conditions [56]. In essence, traditional optimization algorithms are the

unsung heroes in the realm of machine learning. They are the forces that drive the learning process, steering models through the daunting yet fascinating landscape of parameter space. Their role is pivotal, as they underpin the performance and efficacy of machine learning models across various applications. As machine learning continues to surge forward, evolving and expanding in scope, the innovation and refinement of traditional optimization algorithms remain at the forefront, ensuring that models are not just learning but also evolving, becoming more adept at making predictions that are both accurate and reliable. The interplay between traditional optimization algorithms and machine learning is a testament to the synergy between mathematical precision and data-driven insights. It's a partnership that underlines the journey of machine learning models from nave learners to sophisticated predictors, capable of unraveling complex patterns and making sense of the world through data. As this journey unfolds, the role of traditional optimization algorithms becomes ever more crucial, highlighting their enduring relevance in a rapidly advancing field. Their contribution is not just in shaping models but in shaping the future of machine learning, paving the way for innovations that will redefine the boundaries of what machines can learn and achieve [56].

4.2 Methodological Framework: Implementing Artificial Neural Networks with Traditional Optimization Algorithms

The methodology of integrating Artificial Neural Networks (ANN) with traditional optimization algorithms involves a structured approach to learning, where the network iteratively adjusts its weights to minimize the discrepancy between its predictions and the actual data. This process encompasses two primary phases: forward propagation, where inputs are passed through the network to generate predictions, and backward propagation, where the network adjusts its weights in response to the error in its predictions. Below, the methodology is outlined in a formal manner, adhering to the structure and style requested:

4.3 Methodology for Integrating ANN with Traditional Optimization Algorithms

1. **Initialization of Weights:**

   - Initiate the weights w_{ji} connecting the j^{th} hidden neuron z_j to the i^{th} input x_i, and the weights w_{lj} connecting the l^{th} output neuron y_l to the j^{th} hidden neuron z_j. These weights are randomly assigned to begin the learning process.
   - The number of input nodes is denoted by n, the number of neurons in the hidden layers by j, and the number of output neurons by l.

2. **Forward Propagation:**

   - Calculate the input of the j^{th} hidden neuron (net_j) as a weighted sum of the inputs: $net_j = \sum_{i=1}^{n} w_{ji} x_i$.
   - Apply an activation function f_h to this weighted sum to obtain the output of the j^{th} hidden neuron $z_j = f_h\left(\sum_{i=1}^{n} w_{ji} x_i\right)$.
   - Compute the output of the network y^* by applying another activation function f_o to the weighted sum of the hidden layer outputs: $y^* = f_o\left(\sum_{j=1}^{J} w_{lj} z_j\right)$.

3. **Calculation of Error:**

   - After obtaining the output from the forward propagation phase, compare the network's predicted output y^* with the actual output y. Calculate the error E using the formula: $E = \frac{1}{2}\sum_{j=1}^{l}(y_{ij} - y_{ij}^*)^2$, where i ranges from 1 to m, representing the number of input samples.

4. **Backward Propagation (Weight Update):**

   - If the calculated error E exceeds a predefined tolerance level, initiate the backward propagation phase. In this phase, the error term is propagated back through the network, allowing for the adjustment of weights.
   - This process involves computing the gradient of the error with respect to each weight and then adjusting the weights in the direction that most reduces the error. This adjustment is typically done using traditional optimization algorithms such as Gradient Descent or its variants.

- The weights are updated iteratively, with each iteration aiming to reduce the overall error. The learning rate, a hyperparameter, determines the size of the weight adjustments. Careful tuning of the learning rate is crucial as it influences the convergence of the learning process.

5. **Iteration and Convergence:**

   - The process of forward propagation, error calculation, and backward propagation is repeated iteratively. With each iteration, the weights of the network are adjusted in a way that the error between the network's output and the actual data is minimized.
   - The iteration continues until the error E is reduced to an acceptable level, or a predefined number of iterations are completed. The goal is to reach a point where the network's predictions are sufficiently accurate, indicating that the model has learned the underlying patterns in the data.

By following this structured approach, Artificial Neural Networks can effectively learn from data. The integration of traditional optimization algorithms in the weight update phase is pivotal, as it guides the network towards optimal performance by minimizing the prediction error through systematic and efficient weight adjustments. This methodology underpins the learning capability of ANNs and is fundamental to their application in various predictive modelling tasks.

4.4 Traditional Algorithms

4.4.1 Gradient Descent Algorithm

Gradient Descent is a first-order iterative optimization algorithm widely used in machine learning and statistics to minimize a function. By iteratively moving towards the steepest descent, the method is employed to find the local minimum of a differentiable function. This approach is particularly prevalent in the training of machine learning models, where the objective is to adjust the model parameters to minimize the loss function, effectively improving the model's predictions on given data.

Fundamentals of Gradient Descent:

The core idea of Gradient Descent is to update the parameters of the function in the opposite direction of the gradient of the objective function with respect to the parameters. The gradient provides the direction of the steepest increase, and moving in the opposite direction is expected to lead to a decrease in the function value. The method is iterative and requires the selection of a step size, also known as the learning rate, which determines the size of the steps taken towards the minimum.

Mathematical Formulation:

Consider a function $f(\theta)$ that needs to be minimized, where θ represents the parameters of the function. The Gradient Descent algorithm updates the parameters θ as follows:

$$\theta_{\text{new}} = \theta_{\text{old}} - \alpha \cdot \nabla f(\theta_{\text{old}}) \tag{4.1}$$

where α is the learning rate, and $\nabla f(\theta_{\text{old}})$ is the gradient of the function f at θ_{old}.

Learning Rate Selection:

The learning rate α is a crucial hyperparameter in Gradient Descent. If it's too small, the algorithm will converge slowly, while a too large learning rate might lead to divergence or oscillation around the minimum. Proper tuning of the learning rate is vital for the performance and convergence of the algorithm.

Variants of Gradient Descent:

- Batch Gradient Descent: Computes the gradient using the entire dataset. It's guaranteed to converge to the global minimum for convex error surfaces and to a local minimum for non-convex surfaces.
- Stochastic Gradient Descent (SGD): Computes the gradient using a single sample at each iteration. This variant is typically faster and can lead to better solutions in cases of non-convex error functions.

- Mini-batch Gradient Descent: A compromise between batch and stochastic variants, it computes the gradient against a subset of the data at each step. This approach aims to balance the convergence speed and stability.

Applications and Challenges:

Gradient Descent is fundamental in machine learning for training models, especially in linear regression, logistic regression, and neural networks. Despite its widespread use, Gradient Descent faces challenges such as choosing an appropriate learning rate, avoiding local minima in non-convex functions, and computational inefficiency with large datasets.

In conclusion, Gradient Descent is a pivotal optimization technique in machine learning and statistical modelling. Its concept of iteratively moving towards the steepest descent forms the basis for many algorithms in finding the minimum of a function. The method's simplicity and versatility make it an indispensable tool in the field, albeit with considerations and challenges that require careful parameter tuning and algorithm selection based on the specific problem and data at hand.

4.4.2 Gradient Descent with Momentum

Gradient Descent with Momentum is an advanced optimization algorithm that improves upon the classical Gradient Descent method by considering the 'momentum' of the parameters. It's designed to accelerate the convergence, especially in the context of high-dimensional data and non-convex optimization landscapes. By incorporating momentum, the algorithm aims to navigate more effectively through ravines and flat regions in the loss function landscape. This section provides a detailed examination of the Gradient Descent with Momentum method, emphasizing its theoretical basis, mathematical formulation, and practical implications.

Fundamentals of Gradient Descent with Momentum:

Gradient Descent with Momentum takes into account not only the current gradient but also the previous gradients historically to determine the direction of the next step. The 'momentum' term increases for dimensions whose gradients point in the same direction

and reduces updates for dimensions whose gradients change directions. This leads to faster convergence and reduced oscillation.

Mathematical Formulation:

Consider a function $f(\theta)$ that needs to be minimized, where θ represents the parameters of the function. In Gradient Descent with Momentum, the parameters are updated as follows:

$$\begin{aligned} \upsilon_t &= \gamma \upsilon_{t-1} + \alpha \nabla f(\theta_{t-1}), \\ \theta_t &= \theta_{t-1} - \upsilon_t. \end{aligned} \tag{4.2}$$

where:

- υ_t is the velocity at time t.
- γ is the momentum coefficient, typically set between 0.9 and 0.99.
- α is the learning rate.
- $\nabla f(\theta_{t-1})$ is the gradient of the function at θ_{t-1}.

Role of Momentum:

The momentum term γ determines the contribution of the previous velocity to the current velocity. A high momentum term accelerates the convergence towards the consistent direction and dampens the oscillations in the updates. The velocity term υ_t accumulates the gradient elements, considering the historical gradients and smoothing out the updates.

Learning Rate and Momentum Selection:

The selection of the learning rate α and momentum term γ is crucial for the performance of the algorithm. While a suitable learning rate ensures that the steps towards the minimum are of appropriate size, the momentum term helps in navigating the landscape more effectively, avoiding local minima and oscillations.

Applications and Practical Considerations:

Gradient Descent with Momentum is widely used in training deep neural networks, where the landscape of the loss function is complex with numerous local minima and flat regions. The method is known for its fast convergence and stability compared to the standard Gradient Descent, especially in the context of complex optimization problems encountered in machine learning.

In conclusion, Gradient Descent with Momentum is a significant enhancement of the classical Gradient Descent optimization method. By integrating the concept of momentum, it addresses the limitations of Gradient Descent, such as slow convergence and susceptibility to local minima. The method's ability to accumulate gradients from the past and adjust the updates directionally makes it a powerful tool in machine learning, particularly in the training of sophisticated models where the optimization landscape is challenging. As with any optimization technique, the key to its successful application lies in the careful tuning of its parameters and understanding the problem's specific characteristics.

4.4.3 Root Mean Square Propagation (RMSProp) Method

RMSProp (Root Mean Square Propagation) is an adaptive learning rate optimization algorithm designed to address some of the shortcomings of the Gradient Descent algorithm, particularly in the context of training deep neural networks. Developed by Geoff Hinton, RMSProp modifies the learning rate for each parameter by dividing the learning rate for a weight by a running average of the magnitudes of recent gradients for that weight. This method aims to resolve the radically diminishing learning rates in Adagrad by utilizing a moving average of squared gradients to scale the learning rate, allowing for a more controlled and efficient convergence of the model.

Fundamentals of RMSProp:

RMSProp adjusts the learning rate for each parameter based on the recent magnitudes of the gradients for that parameter. The algorithm maintains a moving average of the square of gradients and divides the gradient by the square root of this average to scale the learning rates.

Mathematical Formulation:

Given a differentiable function $f(\theta)$ that needs to be minimized, where θ represents the parameters of the function, RMSProp updates the parameters as follows:

1. Calculate the gradient: $g_t = \nabla f(\theta_{t-1})$, which is the gradient of the function f with respect to the parameters at iteration $t-1$.
2. Update the moving average of the squared gradients: $E_t = \beta E_{t-1} + (1-\beta) g_t^2$, where E_t is the moving average at time t, and β is the decay rate.
3. Update the parameters: $\theta_t = \theta_{t-1} - \frac{\alpha}{\sqrt{E_t + \varepsilon}}$, where α is the learning rate, and ε is a small constant added to improve numerical stability.

Advantages of RMSProp:

RMSProp overcomes the decreasing learning rate issue in Adagrad, ensuring that the learning rates do not diminish too rapidly, which facilitates more stable and effective convergence. The algorithm is computationally efficient and straightforward to implement, making it suitable for high-dimensional optimization problems.

Parameter Selection and Practical Considerations:

The choice of the learning rate α and the decay rate β is critical for the performance of RMSProp. These parameters need careful tuning for optimal results. The constant ε is generally set to a small value (e.g., 10^{-8}) to maintain numerical stability. As an adaptive method, RMSProp usually requires less manual tuning of the learning rate and can be more robust across different model architectures and datasets.

In conclusion, RMSProp presents a robust and efficient approach to optimizing complex models, particularly in machine learning scenarios with large parameter spaces and challenging optimization landscapes. Its per-parameter adaptive learning rate adjustment mechanism addresses the limitations of traditional Gradient Descent, promoting faster and more stable convergence in various applications.

4.4.4 Novel Weight Updating Optimization Algorithm

The Novel Weight Updating Optimization Algorithm represents a sophisticated method for optimizing neural network parameters, specifically focusing on the dynamic adjustment of learning dynamics for weights and biases based on the distinct properties of their respective gradients. This innovative approach is informed by established algorithms that adapt the gradient updates using historical information, thereby providing a nuanced strategy to handle varying gradient magnitudes and behaviors effectively.

Core Principle of the Algorithm:

The algorithm operates on the principle of dynamically modulating the effective scaling of gradient updates, contingent on the specific attributes of the gradients of weights and biases. For weights, the algorithm employs a ratio involving the squared gradient and its logarithmic magnitude. This technique allows for a refined scaling of the gradient, ensuring that the parameter update is sensitive to the gradients magnitude and direction. For biases, the algorithm introduces an exponential component based on the magnitude of the gradient. This addition is intended to amplify the effect of biases in the learning process, recognizing their pivotal role in the neural networks performance.

Weight and Bias Updating Equations:

The weight updating mechanism is governed by the following equations:

$$S_w(t) = \beta \cdot \left| \frac{(dw_i)^2}{\ln(|dw_i|)} \right| \tag{4.3}$$

$$w_{i+1} = w_i - \frac{\alpha \cdot dw_i}{\sqrt{S_w(t) + \varepsilon}} \tag{4.4}$$

Bias updating equation

$$S_b(t) = \beta \cdot e^{|db_i|} \tag{4.5}$$

$$b_{i+1} = b_i - \frac{\alpha \cdot db_i}{\sqrt{S_b(t) + \varepsilon}} \tag{4.6}$$

In these equations, β represents the momentum hyperparameter, crucial for determining the influence of the previous velocity vector, commonly set around 0.08. dw_i and db_i denote the gradients of the loss function concerning the weights w_{i+1} and biases b_{i+1}

respectively. Here, α is a constant learning rate, dictating the base magnitude of the parameter updates. ε is a small constant introduced to ensure numerical stability and prevent division by zero.

Adaptive Gradient Scaling and Parameter Updates:

The algorithm ingeniously divides the update process into two distinct phases: the computation of adaptively scaled gradients and the application of parameter updates utilizing these scaled gradients. While the learning rate α remains constant, the scaling of the gradients using $Sw(t)$ and $Sb(t)$ effectively adjusts the magnitude of updates, thereby mimicking an adaptive-like behavior. This precision holds the promise of enhancing convergence speed and potentially elevating overall model performance.

Distinctive Treatment of Weights and Biases:

The algorithms use of a natural logarithm function for scaling the weights gradients serves to highlight smaller gradients, thereby instilling stability in the update process. Conversely, the application of an exponential function for the biases gradients aims to accentuate their influence, ensuring that the updates are substantial and impactful. This deliberate and adaptive scaling caters to the different roles and sensitivities of weights and biases in the training of neural networks, offering a balanced and effective approach to parameter optimization.

In conclusion, the Novel Weight Updating Optimization Algorithm is a testament to the evolution of parameter optimization techniques in neural networks. By providing a differentiated and adaptive treatment of weights and biases through gradient scaling while keeping the learning rate constant the algorithm sets a new standard for stability, convergence, and overall model performance. Its thoughtful integration of logarithmic and exponential scaling functions aligns the learning process closely with the intrinsic properties of gradients, marking a significant advancement in the field of neural network training and optimization.

4.5 Enhancing Classification Performance: Employing Artificial Neural Networks with Traditional Optimization Algorithms

In this segment of the study, a meticulous image processing approach was employed to extract critical features from a designated dataset of medical images. This comprehensive feature extraction process utilized a trio of advanced techniques: Local Binary Pattern (LBP), Grey-Level Co-occurrence Matrix (GLCM), and Gabor filters. Each technique contributed uniquely to the feature set, with LBP yielding two distinct attributes, GLCM providing five attributes, and Gabor filters contributing a pair of attributes. The extracted features were methodically organized and stored in a '.csv' file, facilitating structured and accessible data handling.

For the purpose of analysis and model training, the dataset was systematically partitioned, allocating 80% of the data for training and reserving the remaining 20% for testing purposes. The focal point of the classification process was an Artificial Neural Network (ANN), which was subjected to an intensive training regimen. This training was conducted using a suite of weight adjustment algorithms, each offering a distinct approach to optimization: the classic Gradient Descent, Gradient Descent augmented with momentum, RMSprop optimization, and a pioneering novel optimization algorithm specifically developed for this study.

Post-training, the ANN showcased its capability to proficiently classify the images into benign and malignant categories. The architecture of the ANN was thoughtfully designed, encompassing an input layer to receive the features, an intermediate hidden layer for processing, and an output layer that produced a range of values between 0 and 1. The activation functions for the hidden and output layers were judiciously selected to optimize the performance of the network. A binary classification was effectively realized by implementing a decision threshold, set at the average of the output neuron values. Consequently, the output was designated as '0' if the resulting hypothesis fell below the threshold, and '1' if it surpassed the threshold.

To achieve and ascertain the highest level of classification accuracy, the study engaged in multiple experimental iterations. The robustness and reliability of the model were further reinforced by employing the k-fold cross-validation method. This method is instrumental in validating the performance of the model across diverse subsets of the dataset,

thereby providing a more comprehensive and dependable evaluation of the model's generalization ability.

In conclusion, this segment underscores the synergy between advanced feature extraction techniques and sophisticated neural network architectures. The meticulous approach to training and validation ensures that the model not only performs with high accuracy but also maintains consistency and reliability, a crucial aspect in the realm of medical imaging and diagnostics.

4.6 Experiments and Results

In this section, a detailed account is provided of the extensive experiments conducted to evaluate the efficacy of various activation functions and optimization algorithms, with a special focus on the novel weight updating method. These experiments were meticulously designed to assess the performance of the algorithms on the classification of breast cancer histopathology images at different magnifications (40X, 100X, 200X, and 400X) using the BreakHis dataset. The evaluation was structured around a robust $k = 5$ cross-validation approach, ensuring the reliability and generalizability of the results. The performance of each model configuration was methodically quantified and compared using a series of meticulously crafted tables, offering a comprehensive view of their effectiveness.

4.6.1 Evaluation Criteria and Methodology

The performance of the models was evaluated using a comprehensive set of metrics, including Precision (P), Recall (R), Specificity (S), F1 Score, and Accuracy. These metrics collectively offer a multifaceted evaluation of the model's classification performance, capturing its ability to identify positive cases accurately, its precision in classification, and its overall accuracy in distinguishing between classes. Additionally, the computational efficiency of each model was also scrutinized, measured in terms of the total time taken for the computation (in seconds) and the number of iterations required for the model to converge. This dual approach in assessment ensures that the models are not only effective in classification but also efficient in computation, an essential consideration for real-world applications [57–60].

4.6.2 Comparative Analysis at Varied Magnifications (40X, 100X, 200X, and 400X)

The comprehensive evaluation at different magnifications, starting from 40X and extending to 100X, 200X, and 400X, showcased the exceptional performance of the proposed novel algorithm across all levels of image detail. At the foundational 40X magnification, the algorithm achieved unparalleled success, registering perfect scores in Precision, Recall, Specificity, F1 Score, and Accuracy across various activation functions. This remarkable achievement was particularly pronounced with the Exponential ReLU + Sigmoid and Leaky ReLU + Sigmoid activation functions, where the algorithm not only demonstrated superior classification prowess but also exhibited enhanced computational efficiency, necessitating fewer iterations for convergence compared to other configurations.

As the magnification level escalated, the robustness and adaptability of the novel algorithm were further underscored. At 100X, 200X, and 400X resolutions, it consistently maintained perfect scores, effectively capturing and classifying the intricate histopathological patterns characteristic of breast cancer tissues. This consistency in high performance, irrespective of the magnification level, signifies the algorithm's ability to handle increased data complexity and detail with remarkable proficiency.

The performance of other optimization algorithms such as RMSprop, Gradient Descent, and Gradient Descent with Momentum also displayed commendable outcomes. However, it was evident that the choice of activation function played a significant role in influencing their success. Notably, configurations incorporating the Exponential ReLU + Sigmoid activation function demonstrated substantial performance enhancement, particularly at higher magnifications. This observation suggests a potential synergy between the activation function and the optimization algorithm, offering a pathway for future optimization and model refinement.

In essence, the comparative analysis across varied magnifications revealed not only the superior performance and efficiency of the novel algorithm but also highlighted the importance of activation function choice in optimizing the performance of traditional algorithms. The findings from this multifaceted analysis provide valuable insights into the dynamics of model behavior in relation to image resolution, laying a solid foundation for future explorations in model optimization and algorithm development.

TABLE 4.1: With 40X Resolution on BreakHis Dataset

Activation	Algorithms	P	R	S	F1 score	Accuracy	Time(sec)	Iterations
Sigmoid	Proposed Novel	1	1	1	1	1	16.57	5000
	RMSprop	0.9662	0.9861	0.9265	0.9760	0.9669	16.36	
	Gradient Decent	0.9955	0.9854	0.9906	0.9904	0.9870	17.97	
	Gradient Decent with Momentum	0.9933	0.9847	0.9861	0.9889	0.985	16.39	
Tanh	Proposed Novel	1	1	1	1	1	19.69	7000
	RMSprop	1	1	1	1	1	13.36	
	Gradient Decent	0.9977	0.9946	0.9956	0.9959	0.9945	21.021	
	Gradient Decent with Momentum	0.9713	0.9190	0.9415	0.9665	0.9554	21.46	
Exponential ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	7.04	2500
	RMSprop	0.9777	0.9820	0.9493	0.9798	0.9719	6.15	
	Gradient Decent	0.9963	0.9921	0.9922	0.9942	0.9920	6.27	
	Gradient Decent with Momentum	0.9955	0.992	0.9906	0.9938	0.9915	5.32	
ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	8.91	5000
	RMSprop	0.9962	0.9877	0.9924	0.9919	0.9890	10.59	
	Gradient Decent	0.9337	0.9971	0.7968	0.9600	0.9338	10.35	
	Gradient Decent with Momentum	0.9978	0.9971	0.9952	0.9974	0.9965	9.84	
Leaky ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	5.37	3000
	RMSprop	0.9786	0.9850	0.9513	0.9817	0.9744	6.93	
	Gradient Decent	0.9970	0.9919	0.9940	0.9944	0.9925	5.14	
	Gradient Decent with Momentum	0.9963	0.9893	0.9922	0.9927	0.9900	6.42	

TABLE 4.2: With 100X Resolution on BreakHis Dataset

Activation	Algorithms	P	R	S	F1 score	Accuracy	Time(sec)	Iterations
Sigmoid	Proposed Novel	1	1	1	1	1	16.7	6000
	RMSprop	1	0.9594	1	0.9793	0.9721	17.06	
	Gradient Decent	1	0.9860	1	0.9929	0.9904	17.6	
	Gradient Decent with Momentum	1	0.9973	1	0.9986	0.9981	16.59	
Tanh	Proposed Novel	1	1	1	1	1	31.92	10000
	RMSprop	1	0.9993	1	0.9997	0.9925	32.24	
	Gradient Decent	0.9993	0.9987	0.9986	0.9990	0.9986	26.9	
	Gradient Decent with Momentum	0.8748	0.8731	0.7512	0.8736	0.8322	28.55	
Exponential ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	9.82	4500
	RMSprop	0.9843	0.9696	0.9614	0.9767	0.9674	11.79	
	Gradient Decent	1	0.9987	1	0.9993	0.9990	10.15	
	Gradient Decent with Momentum	1	0.9987	1	0.9993	0.9990	11.09	
ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	7.41	3000
	RMSprop	0.9852	0.9598	0.9661	0.9722	0.9616	7.5	
	Gradient Decent	0.9275	0.9481	0.7932	0.9345	0.9020	5.33	
	Gradient Decent with Momentum	0.8000	0.7993	0.9982	0.7997	0.8537	6.36	
Leaky ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	5.92	2500
	RMSprop	0.9821	0.8660	0.9636	0.9192	0.8957	5.08	
	Gradient Decent	0.9993	0.9973	0.9984	0.9983	0.9976	5.28	
	Gradient Decent with Momentum	0.9882	0.9719	0.9786	0.9798	0.974	4.44	

TABLE 4.3: With 200X Resolution on BreakHis Dataset

Activation	Algorithms	P	R	S	F1 score	Accuracy	Time(sec)	Iterations
Sigmoid	Proposed Novel	1	1	1	1	1	15.68	6000
	RMSprop	1	0.9364	1	0.9668	0.9553	16.50	
	Gradient Decent	1	0.9931	1	0.9965	0.9950	10.09	
	Gradient Decent with Momentum	1	0.9896	1	0.9947	0.9926	16.96	
Tanh	Proposed Novel	1	1	1	1	1	7.94	3000
	RMSprop	1	0.9877	1	0.9938	0.9915	8.43	
	Gradient Decent	1	0.9959	1	0.9979	0.9970	5.95	
	Gradient Decent with Momentum	1	1	1	1	1	8.43	
Exponential ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	4.17	1500
	RMSprop	0.9283	0.8615	0.8629	0.8934	0.8605	4.97	
	Gradient Decent	1	0.9917	1	0.9958	0.9940	3.42	
	Gradient Decent with Momentum	1	0.9937	1	0.9968	0.9955	4.98	
ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	3.60	2000
	RMSprop	0.9647	0.9013	0.9284	0.9316	0.9086	4.88	
	Gradient Decent	1.0000	0.9916	1	0.9958	0.9940	3.42	
	Gradient Decent with Momentum	0.8000	0.7986	1	0.7993	0.8612	4.28	
Leaky ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	4.33	2000
	RMSprop	0.9622	0.8808	0.9251	0.9192	0.8932	4.63	
	Gradient Decent	0.9804	0.9515	0.9654	0.9651	0.9557	2.44	
	Gradient Decent with Momentum	0.9785	0.9523	0.9574	0.9645	0.9538	5.20	

TABLE 4.4: With 400X Resolution on BreakHis Dataset

Activation	Algorithms	P	R	S	F1 score	Accuracy	Time(sec)	Iterations
Sigmoid	Proposed Novel	1	1	1	1	1	31.62	12000
	RMSprop	0.9992	0.9968	0.9984	0.9980	0.9973	35.63	
	Gradient Decent	0.9975	0.9976	0.9952	0.9975	0.9967	40.17	
	Gradient Decent with Momentum	0.9975	0.9976	0.9952	0.9975	0.9967	38.19	
Tanh	Proposed Novel	1	1	1	1	1	48.08	17000
	RMSprop	1	0.9984	1	0.9992	0.9989	49.55	
	Gradient Decent	0.9975	0.9992	0.9952	0.9984	0.9978	42.71	
	Gradient Decent with Momentum	0.9308	0.9092	0.8748	0.9190	0.8984	56.65	
Exponential ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	11.67	4500
	RMSprop	0.9967	0.9856	0.9931	0.9911	0.9879	11.48	
	Gradient Decent	0.9992	0.9959	0.9984	0.9975	0.9967	14.42	
	Gradient Decent with Momentum	0.9992	0.9975	0.9984	0.9983	0.9978	11.94	
ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	20.57	10000
	RMSprop	0.9992	0.9984	0.9984	0.9988	0.9984	20.84	
	Gradient Decent	0.9308	0.9976	0.7968	0.9573	0.9297	19.06	
	Gradient Decent with Momentum	0.9359	0.9968	0.7986	0.9607	0.9346	12.65	
Leaky ReLu + Sigmoid	Proposed Novel	1	1	1	1	1	14.37	7000
	RMSprop	0.9968	0.9927	0.9932	0.9947	0.9929	13.22	
	Gradient Decent	0.9992	0.9968	0.9984	0.9980	0.9973	16.14	
	Gradient Decent with Momentum	0.9992	0.9976	0.9984	0.9984	0.9978	9.12	

4.6.3 Benchmarking Against Prior Research

In benchmarking the performance of the proposed novel algorithm against prior research methodologies, a stark distinction is observed. While previous methods, including various CNN (Convolutional Neural Network) architectures and innovative deep

learning approaches, have achieved notable accuracies, the novel algorithm has set a new standard by consistently achieving 100% accuracy across all magnifications. This achievement not only highlights the superior performance of the novel algorithm but also marks a significant advancement in the field of breast cancer histopathology image classification. The algorithm's success in outperforming established methodologies underscores its potential as a transformative tool for medical diagnosis and treatment planning.

TABLE 4.5: The following table lists the work done by other researchers and accuracy obtained on the same data set using different methodologies.

Year	**Reference**	**Methodology**	**Accuracy (%)**			
			40X	**100X**	**200X**	**400X**
2016	[59]	Convolution Neural Network (CNN)	90.4	87.4	85	83
2017	[57]	Structural Deep Neural Network(CSDCNN)	95.80 3.1	96.90 1.9	96.70 2.0	94.90 2.8
2018	[58]	Convolution Neural Network (CNN)	94.40	95.93	97.19	96
2019	[60]	Multiple instances learning	87.8 5.6	85.6 4.3	80.8 2.8	82.9 4.1
2019	[61]	Deep Learning, Transfer Learning, GAN	98.20	98.30	98.20	97.50
2020	[62]	Convolutional Neural Network (CNN)	73.41	76.77	83.22	75.81
2022	[63]	Deep inception and residual blocks	80.80	82.76	86.55	85.80
2023	[64]	VGGIN-Net: Deep Transfer Network	0.9588 0.0033	0.9657 0.0087	0.9500 0.0122	0.9313 0.0034
2023		**Proposed work in the present study**	100	100	100	100

The experiments and results detailed in this section provide a clear and comprehensive view of the superior performance and efficiency of the proposed novel weight updating optimization algorithm. With its unmatched accuracy and computational efficiency, the algorithm holds great promise for clinical applications, potentially revolutionizing early cancer detection and diagnosis. Looking ahead, the exploration of this novel algorithm in conjunction with more complex neural network architectures or its application to other types of medical imaging data could offer new avenues for research and development, further extending its impact and utility in the medical field. In summary, the rigorous and detailed comparative analysis, underpinned by a robust empirical evaluation framework, firmly establishes the novel weight updating optimization algorithm as a leading methodology in medical image classification. Its remarkable performance and efficiency set a new benchmark in the field, paving the way for advanced machine learning applications in medical diagnostics and treatment.

4.7 Conclusion

In this investigation, the focal point was to meticulously assess the performance of a newly introduced optimization algorithm in the domain of breast cancer histopathology image classification. The algorithm's efficacy was rigorously compared with established optimization techniques such as Gradient Descent, RMSprop, and Momentum. A salient aspect of this study was the integration of three sophisticated feature extraction methodologiesGabor filters, Local Binary Patterns (LBP), and Gray Level Co-occurrence Matrix (GLCM)which synergistically contributed to the formulation of a rich, multi-faceted feature space for the neural network models.

The empirical results from this comprehensive analysis were quite compelling. The novel optimization algorithm consistently demonstrated superior performance, achieving an unparalleled classification accuracy of 100%. This exemplary performance was not confined to accuracy alone; it extended to encompass other pivotal performance indicators, including the F1-score, precision, sensitivity, and specificity. These metrics collectively underscore the algorithm's robustness and its adeptness in capturing the intricate and nuanced patterns characteristic of histopathology images.

Significantly, the efficacy of the novel optimization algorithm transcended the boundaries of individual activation functions, indicating its versatility and broad applicability. This observation underscores the algorithm's capacity to function cohesively with various architectural nuances of neural network models, further amplifying its utility and relevance in automated image classification tasks.

The implications of these findings are profound and far-reaching. The novel algorithm, when coupled with the integrated feature extraction methodology, emerges as a formidable tool in the realm of medical diagnostics. Its exceptional accuracy and reliability hold tremendous potential in clinical settings, particularly for the early detection and accurate classification of breast cancer, thereby contributing to improved patient outcomes and healthcare strategies.

Looking ahead, the promising landscape outlined by this study beckons further exploration. The potential applicability of the novel optimization algorithm extends beyond breast cancer histopathology, promising to revolutionize a broader spectrum of medical imaging applications. Delving into these possibilities could unravel new dimensions in medical diagnostics, offering pathways to novel discoveries and innovations. As such,

this study not only presents a significant leap in breast cancer histopathology image classification but also lays down a foundational framework for future research endeavors, setting the stage for transformative advancements in medical imaging and diagnostic methodologies.