2023 International Conference on Sustainable Emerging Innovations in Engineering and Technology (ICSEIET)

A Review of Optimization Algorithms for Training Neural Networks

Animesh Srivastava [1]
Assistant Professor [1]
Department of Computer Science & Engineering
Graphic Era Hill University, Dehradun [1]
er.animesh10@gmail.com [1]

Dr. Bhupender Singh Rawat [2]
Associate Professor
College of Smart Computing
COER University, Roorkee [2]
rawat.bhupender@gmail.com [2]

Gulbir Singh [3]
Assistant Professor, MMICT&BM, M.M. (Deemed to be University), Ambala, Haryana, India.
gulbir.rkgit@gmail.com [3]

Dr. Vivek Bhatnagar [4]
Associate Professor, MMICT&BM(MCA), M.M. (Deemed to be University), Ambala, Haryana
bhatnagar.78@gmail.com [4]

Parveen Kumar Saini [5]
Assistant Professor [5]
Department of CSE, Chandigarh University, Gharuan, Mohali, Punjab [5]
parveen.e13339@cumail.in [5]

Shiv Ashish Dhondiyal[6]
Department of Computer Science & Engineering [6]
Graphic Era (Deemed to be University), Dehradun[6]
shivashish1234@gmail.com[6]

Abstract: The selection of the optimization algorithm (optimizer) is one of the most essential endeavors in Deep Learning and across all categories of Neural Networks. It's a matter of experimenting by making mistakes and learning from them. In this research, we will conduct experiments using 7 of the most common optimization algorithms, namely: Stochastic Gradient Descent, Root Mean Square Propagation, Adaptive Gradient Algorithm, Adadelta, Adaptive Moment Estimation, Adamax, and Nadam on MNIST datasets, to determine which one provides the deep neural network with the highest accuracy and performance. A data scientist will benefit from the insightful analysis provided by this study in selecting the most appropriate optimizer to use while modeling their deep neural network.

Keywords: Optimization Algorithms, Neural Networks, Adam, Deep Learning

I. INTRODUCTION

Within artificial intelligence, neural networks have surfaced as a potent instrument that can profoundly transform numerous sectors and applications. The abovementioned sectors encompass computer vision, natural language processing, and robotics. The human brain's structure and functioning influenced these networks' design and functioning. These networks demonstrate exceptional capabilities in identifying intricate patterns and complex correlations within extensive datasets. However, training neural networks is challenging due to the significant computational requirements and the inherent non-linear nature of these networks [1]. Deep learning is a kind of machine learning used for speech recognition, text classification, and other complex tasks. Deep learning models have hidden layers, loss functions, activation functions, inputs, and outputs. All deep learning algorithms try to generalize the data by using an algorithm and creating predictions based on information that has not yet been observed. In addition to needing an algorithm that optimizes results, we need an algorithm that maps different kinds of inputs to the various types of outputs optimization methods find the parameters (weights) that minimize error when mapping inputs to outputs.

The training procedure of a neural network involves adjusting its parameters, commonly known as weights and biases, to minimize a predetermined objective function, also known as the loss function. The objective function serves as a quantitative metric to assess the neural network's performance. The goal is to determine the optimal parameter configuration that yields the highest level of performance.

The utilization of optimization techniques in this procedure holds significant importance as they dictate how these parameters are modified during the training phase. In the context of training neural networks, the choice of optimization method significantly impacts the efficiency and effectiveness of the training process. Throughout an extended period, researchers have developed and enhanced a diverse range of optimization algorithms, each exhibiting distinct advantages and drawbacks [2].

This comprehensive researchgoalis to explore a selection of essential optimization algorithms commonly utilized in the training of neural networks. This study aims to delve into the mathematical underpinnings and operational principles that underlie each algorithm, to impart a comprehensive understanding of their approach to solving optimization problems. Furthermore, we discussed the pragmatic considerations and optimal approaches that ought to be adhered to when choosing the suitable optimization technique for specific neural network structures and tasks [3].

This paper covered classical algorithms, including SGD, Root Mean Square Propagation, Adaptive Gradient Algorithm, Adadelta, Adaptive Moment Estimation, Adamax, and Nadam on MNIST datasets. As we navigate the intricacies of optimization algorithms for neural network training, it is imperative to acknowledge the ongoing evolution of this field. In response to the escalating intricacy of deep learning models and their extensive application across diverse tasks, novel techniques, and enhancements are continuously developed and implemented.

II. LITERATURE REVIEW

Neural networks are versatile machine learning tools that can be applied to a wide range of applications. However,

979-8-3503-2919-3/23/$31.00 ©2023 IEEE

Authorized licensed use limited to: Universitaet Goettingen. Downloaded on December 02,2025 at 19:50:45 UTC from IEEE Xplore. Restrictions apply.

2023 International Conference on Sustainable Emerging Innovations in Engineering and Technology (ICSEIET) | 979-8-3503-2919-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICSEIET58677.2023.10303287

training neural networks can be challenging, with significant implications for training time and accuracy depending on optimization algorithm selection [19][20].

In this survey of the relevant literature, we survey the various optimization algorithms used to train neural networks.

Mirjalili et al. [4] presentedthe WOA method for optimization. This algorithm was given the name whale optimization. Nature inspired this technique. The algorithm solves the challenge by mimicking whale hunting. Three operators simulate humpback whales' hunting, surrounding, and bubble net-catching. This involves hunting, surrounding, and using bubble nets. WOA evaluation requires 39 actions. Mathematical optimization has 29 problems, while structural design has 6. The study revealed that the WOA technique may be quite successful in real-world settings with uncertain search locations. Its potential for success showed this. WOA trends suggest the same.

Seyedali et al. [5] present a completely novel optimization strategy, which they referred it as the grey wolf optimization (GWO). This strategy was established with the assistance of a wolf pack in mind when it was conceived. This exists in the natural world as well as in the hunting activity that they engage in. The purpose of this experiment is to simulate the hierarchy of leadership in the pack by using four distinct kinds of grey wolves. These wolves are referred to as "alpha," "beta," "delta," and "omega" correspondingly.

Masadehet al. [6] sea lion optimization (SLnO) approach was inspired by nature. This method mimics sea lion hunting in their natural habitat. This study also examined the manipulation phase and convergence of the suggested approach on 23 mathematical optimization problems. Complex situations needed the greatest attention. The approach avoids locally optimal solutions and converges rapidly even with more repetitions.

Seyedali et al. [7] described an ant lion optimizer (ALO). The ALO algorithm scientifically replicates the natural interface between ants and antlions. The approach solves optimization issues including random ant walks, trap development, and capture, prey acquisition, and trap reconstruction. Trap reconstruction is also considered. Ants' unexpected behavior helps prevent local optimal solutions, and the system's few adjustable parameters make it straightforward to implement. Ant colony collapse disorder is serious.

Arora et al. [8] proposed the butterfly optimizations algorithm (BOA) in 2018 to solve global optimizations. The approach simulates butterfly mating and food-seeking behavior. The approach was tested on thirty benchmark test functions and compared to other metaheuristic algorithms. BOA solves optimization problems in several sectors because of its quickness. Its drawbacks include a lower population and a higher risk of "local optimum.

Read et al. [9] examined smart grid fake data injection (FDI) attacks by attack methodology, effect, and target. The power system and smart grid security requirements were utilized to evaluate the cyberattack models found in the study. These criteria estimated model efficacy. This was done to identify model issues, which led to their implementation. To improve smart grid cybersecurity, many FDI attack research topics have been proposed.

Khosravani et al. [10] explored creating a knowledge-based system to improve injection molding. This improved product quality. The suggested method uses cutting-edge technologies to help injection molding firms increase productivity and efficiency. This study focuses on "Simulation and Generative Design," "Additive Manufacturing," and "Virtual Reality" to optimize production. The investigation may improve the cost-effective and appropriate solutions produced by the knowledge-based system. Research and experimentation may grow and experiment.

The comparison table of this literature review is shown in Table 1.

TABLE 1: COMPARISON TABLE OF LITERATURE REVIEW

Research Article	Optimization Method	Inspiration	Key Features	Application/Domain
Mirjalili et al. [4]	Whale Optimization (WOA)	Nature (Whale hunting)	Mimics whale hunting behavior	Mathematical optimization, structural design
Seyedali et al. [5]	Grey Wolf Optimization (GWO)	Nature (Wolf pack)	Simulates leadership hierarchy with wolf types	General optimization
Masadeh et al. [6]	Sea Lion Optimization (SLnO)	Nature (Sea lion hunting)	Mimics sea lion hunting in habitat	Mathematical optimization
Seyedali et al. [7]	Ant Lion Optimizer (ALO)	Nature (Ant-lion interaction)	Replicates ant-lion behavior	Optimization with few adjustable parameters
Arora et al. [8]	Butterfly Optimization (BOA)	Nature (Butterfly behavior)	Simulates butterfly mating and foraging	Global optimization across sectors
Read et al. [9]	Smart Grid FDI Attacks	Cybersecurity	Analyzes fake data injection attacks on smart grid	Smart grid security
Khosravani et al. [10]	Knowledge-based System	Manufacturing tech	Uses modern tech for injection molding improvement	Injection molding optimization

III. OPTIMIZATION ALGORITHMS

When it comes to improving the functionality of a neural network, optimization algorithms, also known as optimizers, are an essential component. They use a standard method to adjust the hyperparameters of a model by the design they have created. The behavior of an optimizer may be affected by hyperparameters such as the learning rate, which determines the optimizer's appriseinstruction [11]. The combination of any optimizer's hyperparameters and update algorithm may be used to tell any two optimizers apart from one another.Our optimization aim is the loss function. The loss function and optimization approach are needed to compile our model. An optimizer must adjust our model's nodes' weights and learning rates to minimize the loss function during training. To summarize, the primary objective of an optimizer is to decrease the amount of error that occurs during training. Now,

Authorized licensed use limited to: Universitaet Goettingen. Downloaded on December 02,2025 at 19:50:45 UTC from IEEE Xplore. Restrictions apply.

before we go on to the Optimizers that we are going to, let's have a little discussion about use in the course of our investigation.

A. Stochastic Gradient Descent (SGD)

The basic optimization technique that is at the core of training neural networks is known as stochastic gradient descent, abbreviated as SGD for short. It is a simple but effective iterative strategy that is used to reduce the impact of the loss function during the learning process. It is computationally efficient and well-suited for huge datasets since the core principle underlying SGD is to update the model's parameters in tiny increments by examining just a random portion of the training data at each iteration [12].

The traditional method of gradient descent calculates the loss function concerning all of the training instances in each iteration. This results in a computationally costly algorithm, particularly when applied to large datasets. In contrast, SGD makes use of randomization by picking a mini-batch (subset) of training samples in a randomized manner to estimate the gradient at each step. SGD avoids local minima and effectively explores parameter space because of its unpredictability.SGD neural network training begins with weight and bias initialization. Training follows. The method adjusts these parameters iteratively to lower the loss function during training. This function quantifies the deviation of anticipated output from ground-truth labels. SGD traverses the training data in mini-batches and adjusts parameters to quickly learn from individual data points and adapt to data distribution changes. Mini-batches do this.

B. Root Mean Square Propagation (RMSprop)

It trains neural networks through adaptive optimization. It solves various limitations of stochastic gradient descent (SGD) by extending it. Setting a learning rate and slow convergence on problem domains are downsides.The fundamental concept underlying RMSprop is to modify the learning rates for the various parameters of the neural network depending on the historical magnitudes of the gradients. This is done to optimize the network's performance. The convergence is sped up as a result of this adaptation, and the odds of finding a better optimal value inside the parameter space are improved. The adjustable learning rates of RMSprop serve to expedite the convergence of the optimization process, particularly in domains where the landscape of the loss function has varied curvature and the gradients could have various magnitudes. This is especially true in situations when RMSprop is used for problems in which the gradients might have different magnitudes. In addition to this, it offers a reliable technique for dealing with gradients that are either noisy or sparse [13].

C. Adaptive Gradient Algorithm (Adagrad)

It isidenticaland comparable to the stochastic gradient descent method. However, Adagrad is not likely to employ adaptive gradients to increase resilience. The inability to manually adjust Adagrad's learning rate is eliminated, which is one of the algorithm's primary benefits. On the other hand, the accumulation of squared gradients in the denominator is the algorithm's major weakness. During the training process, the aggregate total increases as a result of each new term being added positively. This causes the learning rate to progressively decrease until it is no longer visible, and as a result, our algorithm loses its capacity to acquire further information. This deficiency is one that Adadelta hopes to improve upon [14].

D. AdaDelta

The stochastic gradient descent technique known as AdaDelta is another family member that is part of the Adagrad family set.Another adaptive optimization approach that is used in the training of neural networks is called AdaDelta. It is an extension of the RMSprop method that seeks to overcome some of the constraints that were present in the original, most notably the need for the human setting of the learning rate hyperparameter. In the same way, as RMSprop does, AdaDelta adjusts the learning rates for each parameter depending on the previous gradients; however, the technique that it takes is a little bit different. In addition to this, it does not rely on adaptive methods for the tuning of hyperparameters, and it is also remarkably resistant to noisy gradient details. It does this by establishing a fixed size, which in turn acts as a restriction on the aggregate previous gradients [15].

E. Adaptive Moment Estimation (Adam)

It is the greatestgeneral optimizer. RMSprop and Adagrad are closely related. Adam optimizes using the L2 norm or Euclidean norm, which is efficient, adaptable, and fast-convergent. Like RMSprop, it employs squared gradients to scale the learning rate, but like stochastic gradient descent with momentum, it calculates the gradient's moving average rather than the gradient itself. It adjusts network weight learning rates using the first and second moments of gradient estimations. Adam, short for adaptive moment estimation, computes different learning rates for different parameters.

The Adam optimizer can create an adaptive learning rate that can traverse the optimization landscape during training in an efficient manner. This adaptability contributes to quicker convergence, which in turn improves the neural network's overall performance [16].

F. Adamax

In the same study that introduced Adam, Adamax was also shown for the first time. It is possible to refer to it as a version of the Adam optimizer that makes use of either the infinite or max norm for optimization. Adamax will perform better than Adam when given data that is usually noisy in terms of gradient updates [17].

G. Nesterov-accelerated Adaptive Moment Estimation (Nadam)

It is an advanced optimization approach for training neural networks. Nadam optimizes deep learning models using NAG and Adam.Neural networks' large dimensionality and non-convex loss landscapes make optimization difficult. Traditional optimization methods like stochastic gradient descent (SGD) fail to negotiate these complicated surfaces, slowing convergence and risking inferior solutions [18].

IV. Result and Discussion

We used the MNIST dataset to carry out the comparison of several optimization strategies. Handwritten digits make up a significant portion of the MNIST database. It comes with a training set that has 60,000 instances, as well as a test set that contains 10,000 cases. Our model was evaluated using a batch size of 64 for ten epochs. After experimenting with a variety of optimizers, the results that all of them provide in terms of accuracy in training and testing are shown below. Table 2

Authorized licensed use limited to: Universitaet Goettingen. Downloaded on December 02,2025 at 19:50:45 UTC from IEEE Xplore. Restrictions apply.

shows the experimental training and testing accuracy results for seven optimization algorithms, i.e. SGD, RMSprop, Adagrad, Adadelta, Adam, Adamax, and Nadam. The Adam optimizer has the highest accuracy of 98.89% among all of them.

TABLE 2: EXPERIMENTAL RESULTS OF TRAINING AND TESTING ACCURACY FOR DIFFERENT OPTIMIZATION ALGORITHMS

S. No.	Optimizer	Training Accuracy	Testing Accuracy
1.	SGD [12]	94.60%	96.59%
2.	RMSprop [13]	98.78%	98.75%
3.	Adagrad [14]	89.29%	92.70%
4.	AdaDelta [15]	73.72%	83.75%
5.	Adam [16]	99.12%	98.89%
6.	Adamax [17]	98.62%	98.67%
7.	Nadam [18]	99.11%	98.88%

TABLE 3: COMPARISONS OF TRAINING ACCURACY AND TESTING ACCURACY

Average Training accuracy	Average Testing Accuracy
93%	95%

Our first optimizer was used as SGD. The patterns are shown in Figure 1 to Figure 7 emerging from the accuracy plots: the training and testing accuracies stay practically the same for the whole of the training period for the Stochastic Gradient Descent, Root Mean Square Propagation, Adaptive Gradient Algorithm, Adadelta, Adaptive Moment Estimation, Adamax, and Nadamoptimizer respectively.

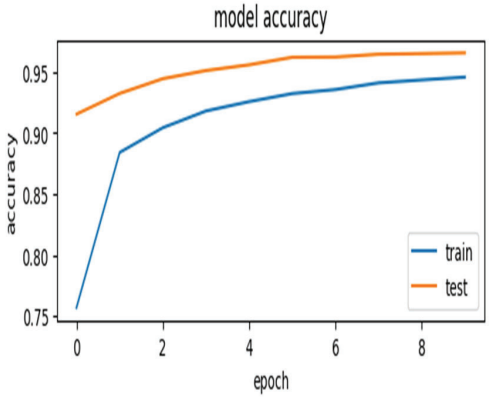

Fig. 1. SGD Optimizer Training and Testing Model Accuracy

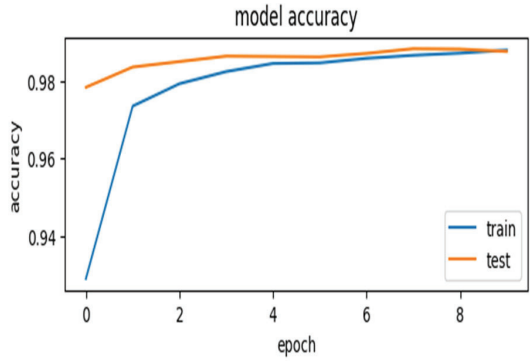

Fig. 2. Figure 2: RMSprop Optimizer Training and Testing Model Accuracy

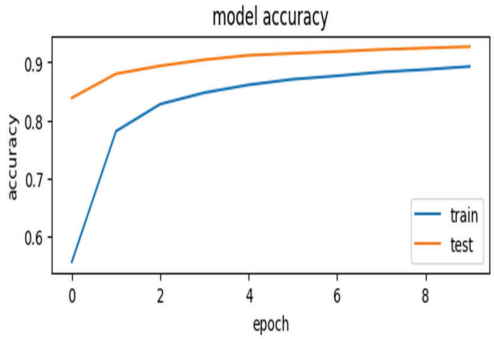

Fig. 3. Adagrad Optimizer Training and Testing Model Accuracy

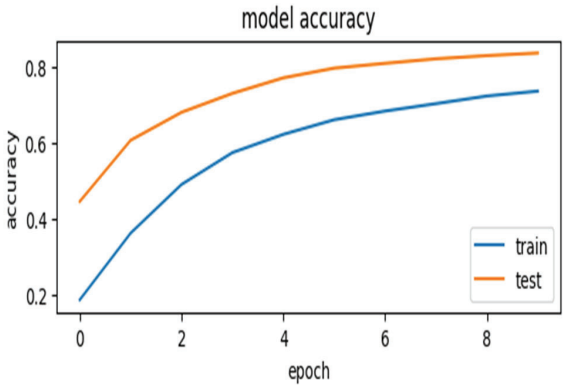

Fig. 4. AdaDelta Optimizer Training and Testing Model Accuracy

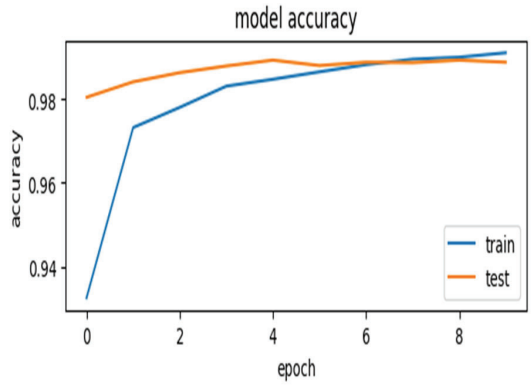

Fig. 5. ADAM Optimizer Training and Testing Model Accuracy

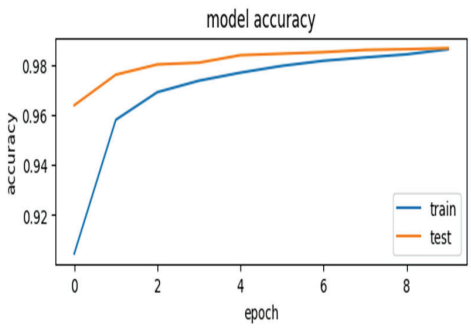

Fig. 6. Adamax Optimizer Training and Testing Model Accuracy

Authorized licensed use limited to: Universitaet Goettingen. Downloaded on December 02,2025 at 19:50:45 UTC from IEEE Xplore. Restrictions apply.

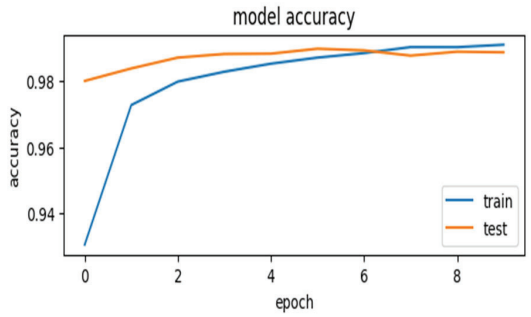

Fig. 7. Nadam Optimizer Training and Testing Model Accuracy

In the preceding experiment, we found several patterns, and as a result, we concluded that the Adam and Nadam optimization Algorithms provide the highest possible accuracy in training and validation.

V. Conclusion

We reviewed several optimization strategies for training neural networks in this thorough study. These methods provide different optimization benefits and drawbacks. After looking at the graphs and doing some trend analysis, we concluded that the 'Adam' Optimization Algorithm works best with Neural Network Models, regardless of the situation. Consequently, it can virtually operate with any classification model, which results in the highest possible accuracy. The selection of the optimal optimization technique depends on a wide range of considerations, such as the size of the dataset, the network design, and the needs of the particular job. In tandem with the development of deep learning as a subject, the landscape of optimization algorithms is also evolving, with new iterations and enhancements being offered on an ongoing basis.

In the end, we conclude that the results of our study are consistent with those of the state of the art. As a result of our analysis, we could also determine which optimizers were the most and least cost-effective.

References:

[1] Ercan, Utku Kürşat, Gizem Dilara Özdemir, Mehmet Akif Özdemir, and Onan Güren. "Plasma medicine: The era of artificial intelligence." Plasma Processes and Polymers (2023): e2300066.

[2] Cong, Shuang, and Yang Zhou. "A review of convolutional neural network architectures and their optimizations." Artificial Intelligence Review 56, no. 3 (2023): 1905-1969.

[3] Choubisa, Manish, and Ruchi Doshi. "Crop protection using cyber-physical systems and machine learning for smart agriculture." In Real-Time Applications of Machine Learning in Cyber-Physical Systems, pp. 134-147. IGI Global, 2022.

[4] Mirjalili, Seyedali, and Andrew Lewis. "The whale optimization algorithm." Advances in engineering software 95 (2016): 51-67.

[5] Seyedali, Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. "Grey wolf optimizer." Advances in engineering software 69 (2014): 46-61.

[6] Masadeh, Raja, Basel A. Mahafzah, and Ahmad Sharieh. "Sea lion optimization algorithm." International Journal of Advanced Computer Science and Applications 10, no. 5 (2019).

[7] Mirjalili, Seyedali. "The ant lion optimizer." Advances in engineering software 83 (2015): 80-98.

[8] Arora, Sankalap, and Satvir Singh. "Butterfly optimization algorithm: a novel approach for global optimization." Soft Computing 23 (2019): 715-734.

[9] Reda, HaftuTasew, Adnan Anwar, and Abdun Mahmood. "Comprehensive survey and taxonomies of false data injection attacks in smart grids: attack models, targets, and impacts." Renewable and Sustainable Energy Reviews 163 (2022): 112423.

[10] Khosravani, Mohammad Reza, Sara Nasiri, and Tamara Reinicke. "Intelligent knowledge-based system to improve injection molding process." Journal of industrial information Integration 25 (2022): 100275.

[11] Chen, Xiangning, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Yao Liu, Hieu Pham et al. "Symbolic discovery of optimization algorithms." arXiv preprint arXiv:2302.06675 (2023).

[12] Beznosikov, Aleksandr, Eduard Gorbunov, Hugo Berard, and Nicolas Loizou. "Stochastic gradient descent-ascent: Unified theory and new efficient methods." In International Conference on Artificial Intelligence and Statistics, pp. 172-235. PMLR, 2023.

[13] Nugroho, Budi, and Anny Yuniarti. "Performance of Root-Mean-Square Propagation and Adaptive Gradient Optimization Algorithms on Covid-19 Pneumonia Classification." In 2022 IEEE 8th Information Technology International Seminar (ITIS), pp. 333-338. IEEE, 2022.

[14] Han, Hong-Gui, Zheng-Lai Lin, and Jun-Fei Qiao. "Modeling of nonlinear systems using the self-organizing fuzzy neural network with adaptive gradient algorithm." Neurocomputing 266 (2017): 566-578.

[15] Sabharwal, Seema, and Priti Singla. "Optimised Machine Learning-based Translation of Indian Sign Language to Text." International Journal of Intelligent Engineering & Systems 16, no. 4 (2023).

[16] Liu, Mingrui, Wei Zhang, Francesco Orabona, and Tianbao Yang. "Adam $^+ $: A Stochastic Method with Adaptive Variance Reduction." arXiv preprint arXiv:2011.11985 (2020).

[17] Pranata, Yuanda F., Rita Magdalena, and Nor KumalasariCaecar Pratiwi. "Optimizer analysis on efficient-net architecture for Alzheimer's classification based on magnetic resonance imaging (MRI)." In AIP Conference Proceedings, vol. 2482, no. 1. AIP Publishing, 2023.

[18] Su, Stephanie SW, and Sie Long Kek. "An improvement of stochastic gradient descent approach for mean-variance portfolio optimization problem." Journal of Mathematics 2021 (2021): 1-10.

[19] Singh, G., Yogi, K.K. (2023). Performance evaluation of plant leaf disease detection using deep learning models. Archives of Phytopathology and Plant Protection, 56(3), 209-233.

[20] A. Srivastava and A. Kumar, "A back propagation NN to optimize the IoT network," 2022 International Conference on Computer Communication and Informatics (ICCCI), 2022, pp. 1-4, doi: 10.1109/ICCCI54379.2022.9740861.

Authorized licensed use limited to: Universitaet Goettingen. Downloaded on December 02,2025 at 19:50:45 UTC from IEEE Xplore. Restrictions apply.