PalArch's Journal of Archaeology of Egypt / Egyptology

COMPARISON OF OPTIMIZATION TECHNIQUES BASED ON GRADIENT DESCENT ALGORITHM: A REVIEW

Saad Hikmat Haji[1], *Adnan Mohsin Abdulazeez*[2]

[1,2]Duhok Polytechnic University, Duhok, Kurdistan Region, Iraq

E-mail: [1]saad.hikmat91@gmail.com, [2]adnan.mohsin@dpu.edu.krd

Saad Hikmat Haji, Adnan Mohsin Abdulazeez. Comparison Of Optimization Techniques Based On Gradient Descent Algorithm: A Review-- Palarch's Journal Of Archaeology Of Egypt/Egyptology 18(4), 2715-2743. ISSN 1567-214x

Keyword: Machine Learning, Deep Learning, Optimization, Gradient Descent, Loss Function,

ABSTRACT

Whether you deal with a real-life issue or create a software product, optimization is constantly the ultimate goal. This goal, however, is achieved by utilizing one of the optimization algorithms. The progressively popular Gradient Descent (GD) optimization algorithms are frequently used as black box optimizers when solving unrestricted problems of optimization. Each iteration of a gradient-based algorithm attempts to approach the minimizer/maximizer cost function by using the gradient's objective function information. Moreover, a comparative study of various GD variants like Gradient Descent (GD), Batch Gradient Descent (BGD), Stochastic Gradient Descent (SGD) and Mini-batch GD are described in this paper. Additionally, this paper outlines the challenges of those algorithms and presents the most widely used optimization algorithms, including Momentum, Nesterov Momentum, Adaptive Gradient (AdaGrad), Adaptive Delta (AdaDelta), Root Mean Square Propagation (RMSProp), Adaptive Moment Estimation (Adam), Maximum Adaptive Moment Estimation (AdaMax) and Nesterov Accelerated Adaptive Moment Estimation (Nadam) algorithms; All of which, will be separately presented in this paper. Finally, a comparison has been made between these optimization algorithms that are based on GD in terms of training speed, convergency rate, performance and the pros and cons.

INTRODUCTION

In real-world training, Deep models efficiently remain one of the most significant challenges for academics and the researchers [1] [2] [3] [4]. The GD optimization algorithm plays a significant role in the training of Machine Learning (ML) and Deep Learning (DL) models. Several new variant algorithms have been developed in recent years to further enhance its efficiency [5] [6] [7] [8]. Machine learning (ML) is an area of computer science that makes it possible for computer systems to understand [9] [10] [11] [12] without being explicitly programmed to perform a specific task. In Machine Learning, adding a cost function allows the machine to find a suitable weight values for results [13]. Deep Learning (DL), on the other hand, is concerned with knowledge retrieval using deep networks [14] [15] [16] [17]. In general, the purpose of optimization is to define the function parameter that makes the solution easier. It is a difficult issue behind many machine learning algorithms [18] [19] [20]. By minimizing the cost function, some optimization algorithms can classify the weights, such as the Gradient Descent (GD) algorithm. [6]. By far, GD is one of the most common optimization algorithms and the most popular method of optimizing neural networks. GD works to find a distinct role at a local minimum. It is used to find the values of the parameters (coefficients) of a function that decrease the loss function as much as possible. [21] [22] [23]. There are also several variants dependent on the GD approach that can be used to maximize the algorithm's performance [13]. This paper tries to provide the reader with insights into the behavior of numerous GD-based optimization algorithms; After that, we will continue by identifying different GD variants, then a brief detail of the challenges during model training and presenting the most popular optimization algorithms, Explaining their drive to address these issues and how this relates to the derivation of their update rules. Finally, the comparison among those GD optimization techniques will be discussed. This study's motivation focuses on such optimization algorithms based on GD in terms of convergent speed and training speed while building a machine learning model.

THEORETICAL BACKGROUND

To find a local minimum of a differential equation, GD is a first-order iterative optimization algorithm. The theory is to take repeated steps in the opposite direction of the gradient at the current stage (or an approximate gradient) of the function because this is considered the steepest descent direction [24]. On the other hand, stepping in the gradient's direction will result in a local maximum of that feature. Then the approach is referred to as gradient ascent [9] [25].

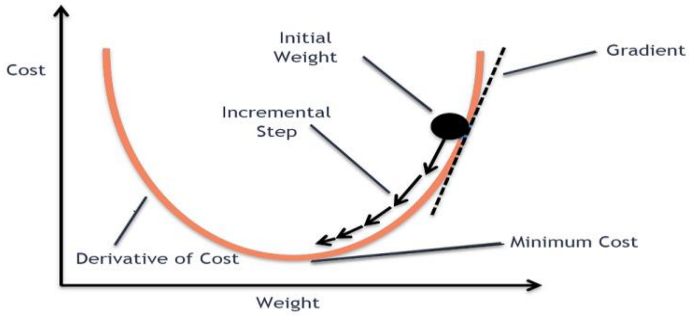

Fig. 1: Gradient Descent Algorithm [26].

Gradient descent variants

There are three variants of GD, which vary in how much data we use to evaluate the gradient of the objective function[27]. Based on the amount of the data[21], we have a trade-off between the update parameter's correctness and the time to execute an update.

Batch Gradient Descent (BGD)

BGD, sometimes named vanilla GD, detects the error inside the training dataset for each example. But the model gets revised after all training examples have been evaluated. This entire process is like a loop, and it is called an epoch of training [28]. Some of the benefits of BGD are its effective calculation, a balanced error gradient and a stable convergence [29]. Some weaknesses are that the steady error gradient can often lead to a convergence condition that is not the best that the model can accomplish. It also demands that the whole training dataset be in memory and available to the algorithm [6]. BGD offers the most precise performance but requires many expensive complete scans of the real data [30].

BGD calculates the gradient of the cost function[21][31] concerning the parameters θ for the whole training dataset, as seen in Equation 1.

$$\theta_{new} = \theta_{old} - \eta . \Delta_{\theta} J(\theta) \qquad (1)$$

Where:
θ_{new} = Next Position
θ_{old} = Current Position
η = Learning Rate (Step Size)
$\Delta \theta J(\theta)$ = Direction of the fastest increase

Stochastic Gradient Descent (SGD)

In optimizing large-scale deep learning models, SGD algorithms have proven to be efficient [32]. The term stochastic means a mechanism or a method connected to a random possibility; therefore, instead of the entire data set for each iteration, a few samples are randomly chosen [28] [33]. SGD aims to find the global minimum by changing the network structure after each training stage [9]. This approach merely reduces the error by approximating the gradient for a randomly chosen batch instead of finding the whole dataset's gradient. In reality, random sampling is done by randomly shuffling the dataset and moving stepwise through batches [34]. SGD executes frequent high-variance changes that allow significant variations in the objective function [35], as seen in Fig. 2.

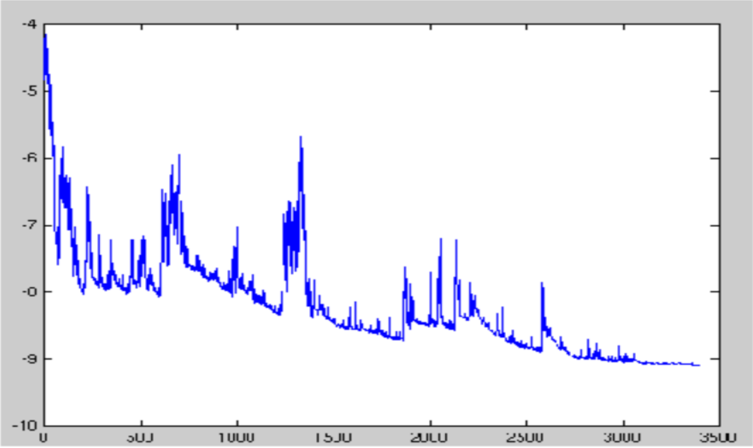

Fig. 2: Stochastic Gradient Descent [21]

Mini-batch gradient descent

Another variation of the principles of SGD and BGD is mini-batch GD. It divides the dataset of training into smaller batches and conducts an update for each of those batches. This provides a balance between SGD robustness and batch GD efficiency [36] [9].

Performance Analysis of Batch GD, SGD, and Mini-batch GD

For several optimization problems, these three GD algorithms perform good, and can all converge to a promising optimum (local or global), but they still suffer from many issues such as the ones mentioned below:

Selection of learning rate

The learning rate η may have a major effect on the convergence of GD algorithms [37] [38]. There is a trade-off between the rate of convergence and overshooting when setting a learning rate. If the learning rate is too high, we could OVERSHOOT the minimum and begin to bounce without meeting the minimum. On the other hand, if the learning rate is too poor, the training could take too long [39] [40].

Adjustment of learning rate

In most cases, for GD algorithms, a fixed learning rate does not work well in the entire updating process [41]. The algorithm will require a greater learning rate in the initial stages to obtain a successful (local or global) rapid optimum. In the latter stages though, the algorithm will need to change the learning rate [39] [38].

Variable individual learning rate

For different variables, instead of the updating process, their upgrade can simply involve a different learning rate. Therefore, it is needed and appropriate to use an individual learning rate for various variables [35].

The Local Minima

Another main challenge is preventing being stuck in their multiple suboptimal local minima by minimizing extremely non-convex error functions common to neural networks [21]. Local minima are the biggest challenge for analyzing convergence [42] [43], as seen in Fig 4.

Gradient descent optimization algorithms

Algorithms of optimization form the foundation on which a machine can benefit from its practice[16] [44] [45]. They measure gradients and try to minimize the function of costs [46]. The learning can be implemented in several ways with various types of optimization methods [47].

Momentum

A common optimization technique is SGD, but when training the algorithm, the runtime is comparatively high. Momentum is designed to learn quickly, especially in the face of wide curvatures, small yet noisy gradients, or stable gradients. The usage of a momentum term is another approach that can assist the network to get rid of local minima [48] [49] [50] [19]. This is, perhaps, the most common extension of the backprop algorithm. Other cases in which this approach is not used are difficult to find. The gradient doesn't point towards the minimum at specific points on the surface, and successive GD steps will oscillate from one side to the other, advancing only very slowly to the minimum [51] [52] [53]. (Fig. 3) illustrates how the incorporation of momentum, by damping these oscillations, tends to drive convergence to the minimal.

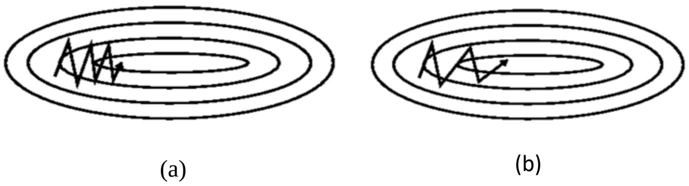

Fig. 3:
(a) SGD without Momentum (b) SGD with Momentum [54].

Nesterov Momentum

Based on parameters that estimate potential positions rather than current parameters, the gradient is calculated. Nesterov Momentum is an improvement over momentum and does not decide the parameter's future location [55] [56]. The Nesterov is a version of the algorithm of momentum inspired by the Nesterov accelerated gradient method. The distinction between this approach and the momentum method is that the velocity is already added to the parameters when calculating the gradient in the Nesterov Method. This can be seen as attempting to incorporate a corrective factor to the conventional momentum form [48].

Adaptive Gradient Descent (AdaGrad)

This is a method that selects the rate of learning according to the situation [57]. As the real rate is calculated from parameters, learning rates tend to adapt. A higher parameter gradient will have a decreased learning rate and vice versa [58]. The theory of AdaGrad is similar to the AdaDelta algorithm in that it measures different learning rates for other parameter elements. Still, it uses gradient squares aggregation: unlike AdaDelta, it uses the moving average of gradient squares. [59].

Adaptive Delta (AdaDelta)

AdaDelta is the AdaGrad extension. AdaDelta works by using several fixed-sized windows instead of accumulating the gradients. It will only monitor the available gradients inside the window [60] [61]. Since the SGD algorithm requires manual learning rate selection, the chosen inappropriate learning rate would lead to low prediction accuracy. However, as an optimization of the SGD algorithm, The Adadelta, known as the adaptive learning rate (LR) algorithm, can automatically adjust the learning rate and increase prediction accuracy. [62].

Root Mean Square Propagation (RMSProp)

One of the most common adaptive stochastic algorithms for Deep Neural Network (DNN) training is RMSProp. [63]. RMSProp modifies Adagrad in a way that it accumulates the gradient [64]. Gradients aggregate into an exponentially weighted average. RMSProp discards past and preserves only current knowledge on the gradient [65] [66].

Adaptive Moment Estimation (Adam)

Adam is a method of SGD optimization that measures adaptable learning rates for each parameter [67]. Adam is one of the most common step-size strategies in the field of neural networks. The name was taken from Adaptive Moments [68]. It's a blend of RMSProp and Momentum. The upgrade operation considers the smooth gradient variant and provides a bias correction mechanism [40] [55]. Adam lowers computing costs, needs less execution memory, and is invariant to gradient diagonal rescaling [69]. RMSprop is a gradient-based optimizer that uses an adaptive learning rate (LR) that varies over time instead of treating the learning rate as a hyperparameter [70].

Maximum Adaptive Moment Estimation (AdaMax)

AdaMax is a form of adaptive SGD, and an Adam version based on the norm of infinity[46]. AdaMax provides the major benefit of being much less sensitive to the option of hyper-parameters relative to the SGD [71]. The AdaMax equation uses the full value of the second momentum component of the ADAM estimation method. This offers a more stable solution [72].

Nesterov-Accelerated Adaptive Moment Estimation (Nadam)

For Adam and RMSProp, this technique is a combination of optimization approaches. It was designed in a similar way to the optimization approach of Adam. The flat momentum, however, was replaced by the momentum of Nesterov. This results in a marked rise in momentum outperformance [73][74].

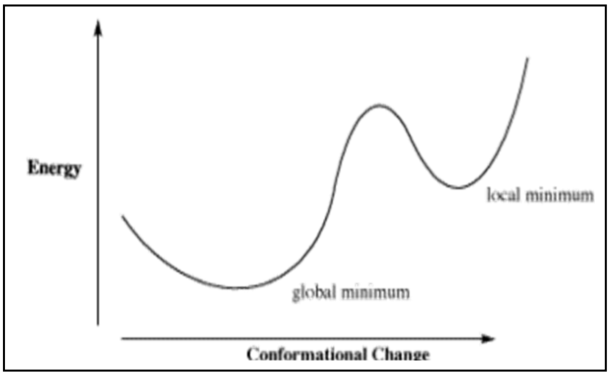

Fig. 4: Global Minima Vs. Local Minima [75]

RELATED WORK

Dogo et al., [28] explained in their research, in a clear "Convolutional Neural Network (ConvNet)" structural setup, A comparative review of the seven most frequently used first-order stochastic gradient-based optimization approaches was performed. that they used three randomly chosen and publicly accessible image classification datasets to test the optimization strategies in terms of convergence rate, precision and error function. The methods tested are the vanilla SGD (vSGD), SGD with momentum (SGDm), with nesterov + momentum (SGDm+n), Adam, RMSProp, AdaDelta, Adaptive Gradient (AdaGrad), Nadam and Adamax. Three datasets have been used which are “Cats and Dogs”, “Natural Images” and “Fashion Mnist”. The findings showed that each optimizer's efficiency differed from each dataset, in contrast to the other optimization strategies, the average experimental results suggested that "Nadam" accomplished better efficiency over the three datasets, while "AdaDelta" did the worst.

Lu and Jin, [5] discussed the problem of improving the performance and classification capability of support vector machines (SVM) based on the algorithm of SGD. Three algorithms of improved SGD have been used to solve this issues which are (Momentum, NAG and RMSprop). The experimental findings reveal that the RMSprop algorithm has a higher convergence velocity and higher accuracy testing for solving the linear SVM.

Lydia and Francis [76] explained some alternatives and hyper-parameters to improve the performance of GD Algorithms such as (Adagrad, Adadelta, RMSProp, Adam and SGD). The Authors explained that the Adagrad optimizer outperforms all other optimizers and strengthens the SGD Algorithm when being trained using different image datasets.

Wang and Ye [33] Showed that in Stochastic Gradient-based optimization algorithms, momentum plays a key role in accelerating or enhancing Deep Neural Network training (DNNs). However, tuning momentum hyperparameters may be a huge computational challenge, so instead of that, they have proposed a new adaptive momentum to enhance the training of DNNs. With this new adaptive momentum, SG eliminates the need for hyperparameter momentum calibration. This raises the learning rate greatly, speeds up DNN teaching, and enhances the final accuracy and robustness of the DNNs being trained. SGD also benefits from adversarial experience with the current adaptive momentum and increases the adversarial robustness of the trained DNNs.

Yaqub et al., [69] The authors address brain tumors as a leading cause of death worldwide, and they connect this outbreak to the challenge of making a prompt tumor diagnosis. To test the efficiency of brain tumor segmentation, the authors compared various optimization algorithms used in the proposed CNN architecture. The gradient-based optimizers used for the comparison were (Nadam, NAG, Adagrad, AdaDelta, SGD, Adam, Cyclic Learning rate, Adamax, and RMSProp) for Convolutional Neural Network. The results found that Adam had the lowest error rate and the highest accuracy rate.

Endah et al., [77] Back-Propagation has issues with the training method, where GD convergence for learning is very poor. And the authors showed that using the adaptive learning rate (LR) and optimizing Momentum could increase the convergence rate. For this study, they used medical records for detecting diabetes. The results showed that for algorithm training, the combination of GD + momentum and adaptive learning rate training algorithm has quicker convergence than a gradient with Momentum or Gradient with an adaptive learning rate.

Wibowo et al., [78] the authors show out that the best optimization algorithm and tuning parameter for neural network (NN) backpropagation was investigated for cancer classification using the microRNA function. The algorithms of optimizations that were used were GD, AdaGrad, Momentum, AdaDelta, RMSProp and Adam. The results of this experiment showed that the highest precision was provided by Adam and RMSProp optimizers, which reached 98.536 percent and 98.54762 percent accuracy respectively.

Solanke et al., [57] discussed that, for intrusion detection, different deep learning methods are used, but they all suffer from certain levels of problems such as high error rate and even increasing the number of iterations to process an output. And this is because the classification system accuracy is low. They proposed a system by using various GD optimization methods such as (Adagrad, Adam, Adadelta and RMSProp) to reduce the rate of errors in the process of training. The results showed that Adam offers much improved outcomes in terms of accuracy, recall and f-measurement.

Fatima [70] discussed the most suitable optimizer for the model of Neural Network. They compared some of the most popular algorithms of optimization with four unrelated datasets to find out which optimizer provides the deep neural network with the highest precision, reliability and performance. In all conditions, Adam optimization algorithm performs well for all four Deep Neural Network models and because of this, it is virtually capable of working with any classification model resulting in the best accuracy. In three out of the four datasets, RMSprop was also considered to be a reasonable option. They also observed that the SGD and Adadelta optimizers struggled to have adequate results in all of the four models as a result of experimentation. Hence, for a supervised DL model, they are the least recommended optimization algorithms.

Tao and Lu, [79] Network-based wind speed forecasting has played a key role in the power grid. In the context of wind speed forecasting, six network parameter optimization algorithms are applied and compared, namely GD, Momentum, AdaGrad, Adam, RMSprop, and Adadelta. The experiment's results suggest that better predictability and much less training time are obtained by the Adam algorithm and RMSprop algorithm than the other optimization algorithms. Nevertheless, for measuring performance, three metrics are used: root mean square error (RMSE), mean absolute error (MAE) and mean absolute percentage error (MAPE).

Lancewicki and kopru [80] Explained that for the training of machine learning & deep learning models, stochastic gradient-based approaches are popular. And discussed that manual hyperparameter correction is very expensive and time-consuming. Therefore, the authors suggested a generic methodology that uses unbiased gradient estimator statistics to change two paramount hyperparameters automatically and simultaneously: the rate, momentum, and learning rate.

YI et al., [6] Discussed that for non-convex issues, most current optimizers may remain at a local minimum right before reaching a global minimum. And inside a complex non-convex system, they have some difficulties identifying the global minimum. The authors developed methods for finding the global minimum value of the non-convex cost function based on Adam to address this problem by introducing a new concept of a non-negligible value at the local minimum. The classical Adam formula has been enhanced, rendering it zero at the global minimum; therefore, the modified optimizer never converges at the local minimum; it can converge only at the global minimum.

Hapsari et al., [81] Argued that data prediction activities entail attempts to boost predictions' accuracy by optimizing parameters in the classification algorithm. Fractional Gradient Descent (FGD) was proposed as an unregulated optimization algorithm for objective functions in an SVM classifier. FGD optimizes the SVM classification model by employing fractional values. With a small learning rate, it has small stages and reaching convergence with smaller iterations in the process of approaching global minimums. The results showed that at iteration 350, the SVM Classifier outputs using SGD optimization hit the convergence stage. With SGD followed by FGD optimization, it reaches a convergence point of 50 iterations shorter than the SVM classifier.

Yu and Liu [82], proposed NWM-Adam (the idea is placing the previous gradients with more memory than the new gradients) This is a new GD optimization algorithm based on a first-order weighting mechanism. To overcome the undesirable convergence behavior of such optimization algorithms that use fixed-sized windows of prior gradients to scale gradient updates and increase the performance of Adam and AMSGrad. The experimental findings indicate that NWM-Adam would outperform other algorithms of optimization.

Le et al., [83] argued that one of the major problems of network defense is the intrusion detection system (IDS). They also discuss that, many of the ML algorithms that are used in IDS like SVM, Neural Network, KNN etc. are still facing some limitations. The authors noticed that, the Nadam optimizer is useful for intrusion detection in the Long Short-Term Memory Recurrent Neural Network (LSTM RNN) model. The results of the experiments revealed that the LSTM RNN model by Nadam optimizer outperforms earlier work. The solution is currently 97.54 percent accurate at performing intrusion detection, 98.95 percent detection rate, and 9.98 percent fair false alarm rate.

Zhao et al., [39] The key optimization technique in deep learning was clarified by the SGD. The consistency of SGD relies heavily on how learning rates change over time. They proposed a new novel, the Energy Index-based Optimization Approach (EIOM), To automatically change the learning rate for backpropagation. The studies showed that the machine learning model based on the EIOM achieved greater classification accuracy than that of the other optimization models, although manual tuning was not necessary other than the choice of a default value. For example, the accuracy rate of EIOM in Convolutional Neural Network (CNN) compared with different gradient descent optimization was 82.1% while for SGD, AdaDelta, RMSProp, Adam and CLR were (76.2 %, 77.9 %, 77.1 %, 79.8 %, 75.5 %) respectively. Table 1 shows the comparison among previous works for different optimization techniques in terms of accuracy.

Hong et al., [87] The novel computational methodology was adopted in the current research which is NN-SGD-GA that incorporated a neural network model trained by an SGD paradigm and a GA that merged the feature selection process and tuning models to construct a landslide susceptibility model. The experiments revealed that the optimized neural network model has the best predictive potential (88.10%), preceded by the Random Forest (RF) (86.26%) and Logistic Regression (LR) (85.83%) models.

Sharma, A [88] despite its simplicity, the SGD approach is an efficient and default primary optimization method for classification models such as NN and LR for machine learning. The researchers suggested a variant of SGD, Directed the SGD, (GSGD) Algorithm attempts in a given dataset to solve this inconsistency by greedily choosing reliable data instances for GD. The guided search with GSGD provides superior convergence and precision rate within a limited time budget than its original counterpart of canonical and other SGDD variants.

Ren et al., [89] by integrating SGD and support vector regression, a data-driven simulation approach for the aero-engine aerodynamic model was proposed (SGDSVR). The simulation data findings and the real flight data prove that the proposed algorithms are stable and accurate in terms of noise and operating conditions.

Nio et al., [90] A new annealed gradient descent (AGD) algorithm was proposed for non-convex Deep learning optimization that could converge at a higher speed to a better local minimum than the standard mini-batch SGD algorithm. The suggested AGD algorithm is used for numerous functions, including image recognition and speech recognition, to train both deep neural DNNs) and Coevolutionary Neural Networks (CNNs). Experimental studies have shown that AGD outperforms SGD vastly in terms of speed of convergency.

Table 1: Summary of Literature Review Related to GD Optimization Algorithms.

Author	**Year**	**Objectives**	**Datasets**	**Results and Accuracy**	**Techniques**
Wang and Ye [33]	2020	Using gradient descent optimization techniques to Improve Deep Neural Networks training and to converges quicker and help us to train DNNs with considerably greater step sizes.	CIFAR10, CIFAR100	SGD reduced error classifications for training ResNet110 for dataset CIFAR10, CIFAR100 from 5.25 percent to 4.64 percent and 23.75 percent to 20.03 percent with this adaptive momentum.	SGD + Adaptive Momentum Optimization techniques
Hong et. Al, [87]	2020	Enhance the accuracy of estimation of vulnerability to landslides	380 landslides and 14 related variables	The experiments revealed that the optimized neural network model has the best	GA, SGD, NN

				predictive potential (88.10%), The Random Forest (RF) (86.26%), and Logistic Regression (LR) (85.83%) models followed it.s	
Yaqub et al. [69]	2020	Measure the efficiency of segmentation of brain tumors by comparing several optimizers based on GD algorithms.	MRI brain image data set, i.e., BraTS2015	Adam has the lowest rate of error and the highest rate of accuracy. Whereas the NAG and RMSProp optimizers failed terribly.	Monte Carlo, Method and optimizers (Nadam, NAG, Adagrad, AdaDelta, SGD, Adam, Cyclic Learning rate, Adamax, RMSProp)
Solanke et al. [57]	2020	Reducing Error Rate in the process of Training by using different optimization algorithms based on gradient descent.	NSL-KDD	The experimental results showed that the optimization algorithm average for Adam is 0.999, RMSProp is 0.98, Adagrad is 0.91 and Adadelta is 0.93	Adagrad, Adadelta, Adam, RMSProp
Nio et. all [90]	2020	Improve speed of convergency and training for Deep Learning	CIFAR-10,	AGD outperforms SGD vastly in terms of speed of convergency.	SGD , CNN

Fatima [70]	2020	Implementing the most effective optimization algorithm for the Neural Network Model (optimizer)	Masters, Toxicity, Workshop, Titanic	Adam Optimizer Accuracy was 92.86 % training and 85% accuracy of validation.	Adam, RMSProp, SGD, Adadelta, Adagrad, AdaMax, Nadam
Lancewicki and kopru [80]	2020	To Minimize the cost function and automatically change the momentum and learning rate	CIFAR10, MNIST	The momentum and learning rate are automatically tuned to mitigate the expected loss maximally using the minibatch statistics.	Adam, AdaGrad, SGD
Ren et. all [89]	2020	a data-driven simulation approach for the aero engine aerodynamic model	Aero-engine flight data	Proposed algorithm SGDSVR are stable and accurate in terms of noise and operational conditions.	SGD, SVR
Hapsari et al., [81]	2020	By using FGD in SVM classifier to increase the precision of prediction models	Rainfall	It can be inferred, from the experimental findings, that FGD uses fraction values to optimize the model of SVM classification such that it has minor measures with a small learning rate of 0.001 and	SVM

				just an error rate of = 0.273070 to reach global minimums and convergence in the 50th iteration that is smaller than SVM-SGDDD	
Lydia and Francis [76]	2019	Improve and enhance the performance of Gradient Descent algorithm by using some alternatives hyper-parameters and optimizers.	MNIST, Caltech-101, COIL-100	The "Adagrad" Optimizer surpasses all other optimizers and improves the Stochastic Gradient Descent Algorithm when training with separate image datasets of distinct nature.	(Adagrad, Adadelta, RMSProp, Adam, SGD)
Yi et al [6]	2019	Solve the problem of reach and stay on local minima instead of global minima by improving ADAM optimizer	Numerical Values	The modified optimizer never converges and can only converge at a global minimum by strengthening the ADAM optimizer by applying a	ADAM, GD, AdaMax

				new term to local minimums with a non-negligible value at local minimum points.	
Yu and Liu, [82]	2019	Solve that undesirable convergence behaviors of some optimization algorithms that use past gradient fixed-sized windows to scale gradient updates and boost Adam and AMSGrad performance.	MNIST, CIFAR-10	The testing findings have shown that new proposed N WM-Adam optimization method performs better on certain convex and non-convex issues throughout the area of ML than other popular GD optimization algorithms.	Momentum, Adagrad, RMSProp, Adam, AMSGrad
Wibowo et al.[78]	2019	Used microRNA data based on optimum precision value, to find the best activation function used in the neural network training method for cancer classification.	National Centre for Commons Genomic Data Institute's Medical Records	The best accuracy provided by Adam and RMSProp, 98.536% achieved by Adam and 98.54762% achieved by RMSProp.	RMSProp, Adam, AdaGrad, Momentum, GD, AdaDelta
Zhao et al.,[39]	2019	Minimizing the cost functionality during the	MNIST, CIFAR-10	Compared with other optimization algorithms,	Different machine learning modules

		process of deep learning training by using a new novel (EIOM)		the experiments show the positive efficiency of the proposed EIOM. For CNN Machine Learning the test accuracy was 82.1 % which is higher percentage than other optimization algorithms used	were used (Logistic Regression, Multilayer perception and CNN). Also, different optimization techniques were used (SGD, AdaGrad, AdaDelta, RMSProp, Adam)
Dogo et al. [28]	2018	The comparative effect of seven optimization algorithms on three image classification datasets was performed using a basic convolution architecture to determine each optimization technique's efficiency.	Three classification image datasets were used which are (Fashion MNIST, Cats and DOGS, Natural Images)	Based on the comparative evaluation among the seven most popular algorithms for optimization It shows that, relative to the other optimization strategies, "Nadam" showed a better and more stable performance in all three datasets analysed with accuracy 85.5 % on “Cats and Dogs”, 91.3 % on “	SGD (Vanilla, with momentum and with nesterov), RMSProp, Adam, Adamax, AdaGrad, AdaDelta and Nadam.

				Natural Images” and 71.2 % on “Fashion MNIST” datasets.	
Sharma, A [88]	2018	Improve the accuracy of SGD, Adadelta, Adam, Momentum, Nesterov and RMSProp	Medical UCI library	GSGD delivers superior convergence and accuracy rate.	SGD, NN, LR
Tao and lu [79]	2018	Wind Speed forecasting in network based by comparing different optimization based on gradient descent algorithms.	wind data set (NingX W)	The RMSprop algorithm and Adam optimization algorithm achieve higher prediction performance and much less training time than other compared algorithms of optimizations algorithms.	AdaGrad, Momentum, RMSProp, Adadelta, Adam
Lu and jin [5]	2017	Based on the SGD algorithm to solve the problem of improving the effectiveness and classification capacity of Support vector machines (SVM).	(Alpha, Gamma, Delta, Mnist, Usps, Letter).	On (Alpha, Gamma, Delta, Mnist, Usps) datasets, the RMSprop-based algorithm for solving the linear support vector machine has higher	RMSprop, Momentum, Nesterov accelerated gradient (NAG)

				convergence speed and better testing accuracy. "Pegasos" on the "Letter" dataset has a higher convergence rate than other techniques.	
Endah et al. [77]	2017	Increase the convergence Rate by using Momentum optimization and Adaptive Learning Rate	Diabetes Medical Records	The results showed that the training of the algorithms by using GD with Momentum optimization and Adaptive Learning Rate has higher convergence	Gradient Descent with Momentum and Adaptive Learning Rate

DISCUSSION

From the related works that has been conducted, it is shown that many researches showed different results while using optimization techniques. If the data set is sparse, one of the adaptive learning-rate strategies could have the best outcomes. An added benefit is that the learning rate would not need to be adjusted, but with the default value, you can generally produce the best results. RMSprop is an expansion of Adagrad that works with its learning speeds, which are radically declining. It is like Adadelta, except that, in the numerator update clause, Adadelta uses the RMS of parameter changes. Adam, eventually, introduces RMSprop bias-correction and momentum. RMSprop, Adadelta, and Adam, in other words, are very similar algorithms that perform well under comparable conditions. Interestingly, several recent papers use vanilla SGD without momentum and a simple annealing plan for the learning rate. As shown, SGD normally manages to find a minimum. Still, it can take much longer than with either optimizer and maybe much more reliant on a reliable initialization and annealing strategy and can get stuck in saddle points rather than local minima. In addition, You should use one of the adaptive learning performance strategies if you think of optimizing efficiency and training a deep or complex neural network. In this paper we surveyed the optimization techniques based on gradient descent and compared those optimization techniques and their effects on training model speed and convergency quality. As shown in table 1, different optimizations algorithms have been used to reduce cost function, increase convergency rate and training speed. Table 2 shows differences among those optimization algorithms that discussed in related work in terms of quality of convergency, speed of training, pros and cons.

Table 2: Comparison between the optimization techniques Categorized by the optimizers

Optimizers	Quality of Convergency	Speed of Training	Pros	Cons	Comment
GD	The quality of convergency for these two optimizers is good	Its medium for the simple model and low for the complex model	Where the objective function is convex, the solution is globally optimal [84].	The cost of the calculation is high [84].	By assigning the correct learning rate, the risk of converging to the local minimum can be handled.
SGD	The quality of convergency for these two optimizers is good	Its medium for the simple model and low for the complex model	For each update, the computation period does not depend on Saved the total number of training samples and a lot of estimation costs [33] [85].	It's challenging to select a reasonable learning rate, and It is not appropriate to use the same learning rate with all dimensions, and the global minimum is difficult to achieve. [33].	By assigning the correct learning rate, the risk of converging to the local minimum can be handled.
Momentum	The convergency quality is	Its fast for the simple	Converges more quickly	For each update, one more	Suitable for a less complicated

	good in this optimizer.	model and medium for the complex model	than the GD algorithm [51].	attribute needs to be calculated [51].	model with less features number.
NAG	Good Quality of convergency	Its fast for the simple model and medium for the complex model	Compared to the GD algorithm, the memory requirement is less [77].	It takes a long time for convergence, sometimes stuck at local minima and also its challenging to select good learning rate [77].	Suitable for a less complicated model with less features number.
AdaGrad	This optimizer misses global Minima	Fast	No need to manually update the learning rate as it varies with iterations adaptively[76].	Due to a large number of iterations, the learning rate would decrease and result in slow convergence [76].	Convenient for a simple quadratic issue
RMSProp	The quality convergency of this optimizer is somehow acceptable	Fast	Growing the inefficient learning challenge at the late stage of AdaGrad. Optimization of non-	The update process may be replicated around the local minimum in the late training period [60].	Suitable for complex model.

			stationary and non-convex problems is sufficient [60].		
ADAM	The quality convergency of this optimizer is somehow acceptable	Fast	Relatively stable is the gradient descent process. With large data sets and high dimensional space, it is suitable for most non-convex optimization problems [86].	In such conditions , the process cannot converge. [86].	Suitable for sparse gradients with a wide number of features on a complex model
AdaMax	The quality convergency of this optimizer is somehow acceptable	Fast	It reduces the need for the learning rate to be manually calibrated. where the scaling of the weights is different , converg	One of AdaGrad's drawbacks is that its method of optimization will result in aggressive learning rates that are monotonically decreasing [74].	Suitable for complex model.

			ence is quicker and more efficient than basic SGD. the scale of the master step does not make it very sensitive [74].		
Nadam	Convergency quality is good here.	Fast	In the Adam algorithm, NADAM integrates Nesterov momentum, which is at times superior to vanilla momentum. Its good optimizer for sparse gradients for complex model[73].	The mechanism may not converge in certain circumstances [86]	Suitable for sparse gradients with a wide number of features on a complex model

CONCLUSION

GD is the most popular optimization algorithm in Machine Learning (ML) and Deep Learning (DL). It is an algorithm for the first-order optimization. This means that only the first derivative is taken into consideration when doing parameter changes. In a specific set of scenarios, machine learning's critical aim is to build a model that works well and provides detailed predictions; However, optimization algorithms are needed to achieve that. Initially, we looked at numerous GD algorithm versions, which are BGD, SGD and Mini-Batch. Then we analyzed optimizers that are most used for optimization: Momentum, Nesterov Momentum, AdaGrad, AdaDelta, RMSProp, Adam, AdaMax and Nadam. We surveyed and reviewed several recent papers and found that each optimizer's output differed concerning learning speed and convergence. Work results show that Nadam showed a more excellent and stable performance in terms of convergence rate, training speed and performance relative to other optimization techniques.

ACKNOWLEDGEMENT

The authors would like to note that, with all its financial resources and encouragement, Duhok Polytechnic University supported this report.

REFERENCES

Zhang, J. (2019). Gradient descent based optimization algorithms for deep learning models training. *arXiv preprint arXiv:1903.03614*.

Zou, T., & Sugihara, T. (2020, November). Fast identification of a human skeleton-marker model for motion capture system using stochastic gradient descent method. In *2020 8th IEEE RAS/EMBS International Conference for Biomedical Robotics and Biomechatronics (BioRob)* (pp. 181-186). IEEE.

Reisizadeh, A., Mokhtari, A., Hassani, H., & Pedarsani, R. (2019). An exact quantized decentralized gradient descent algorithm. *IEEE Transactions on Signal Processing*, *67*(19), 4934-4947.

Maulud, D., & Abdulazeez, A. M. (2020). A Review on Linear Regression Comprehensive in Machine Learning. *Journal of Applied Science and Technology Trends*, *1*(4), 140-147.

Lu, S., & Jin, Z. (2017). Improved Stochastic gradient descent algorithm for SVM. *International Journal of Recent Engineering Science (IJRES)*, *4*(4), 28-31.

Yi, D., Ji, S., & Bu, S. (2019). An Enhanced Optimization Scheme Based on Gradient Descent Methods for Machine Learning. *Symmetry*, *11*(7), 942.

Zebari, D. A., Zeebaree, D. Q., Abdulazeez, A. M., Haron, H., & Hamed, H. N. A. (2020). Improved Threshold Based and Trainable Fully Automated Segmentation for Breast Cancer Boundary and Pectoral Muscle in Mammogram Images. *IEEE Access*, *8*, 203097-203116.

Zeebaree, D. Q., Haron, H., Abdulazeez, A. M., & Zebari, D. A. (2019, April). Trainable Model Based on New Uniform LBP Feature to Identify the Risk of the Breast Cancer. In *2019 International Conference on Advanced Science and Engineering (ICOASE)* (pp. 106-111). IEEE.

Hallen, R. (n.d.). *A Study of Gradient-Based Algorithms*. 26.

Abdulazeez, A. M., Sulaiman, M. A., & Qader, D. (2020). *Evaluating Data Mining Classification Methods Performance in Internet of Things Applications*. *1*(2), 15.

Qader Zeebaree, D., Mohsin Abdulazeez, A., Asaad Zebari, D., Haron, H., & Nuzly Abdull Hamed, H. (2021). Multi-Level Fusion in Ultrasound for

Cancer Detection based on Uniform LBP Features. *Computers, Materials & Continua*, *66*(3), 3363–3382.

Adeen, I. M. N., Abdulazeez, A. M., & Zeebaree, D. Q. (2020). *Systematic Review of Unsupervised Genomic Clustering Algorithms Techniques for High Dimensional Datasets*. *62*(03), 21.

Kamsing, P., Torteeka, P., & Yooyen, S. (2020). An enhanced learning algorithm with a particle filter-based gradient descent optimizer method. *Neural Computing and Applications*, *32*(16), 12789–12800. https://doi.org/10.1007/s00521-020-04726-9

Omar, N., Abdulazeez, A. M., Sengur, A., & Al-Ali, S. G. S. (2020). Fused faster RCNNs for efficient detection of the license plates. *Indonesian Journal of Electrical Engineering and Computer Science*, *19*(2), 974-982.

Zeebaree, D. Q., Haron, H., Abdulazeez, A. M., & Zeebaree, S. R. (2017). Combination of K-means clustering with Genetic Algorithm: A review. *International Journal of Applied Engineering Research*, *12*(24), 14238-14245.

Jahwar, A. F., & Abdulazeez, A. M. (2020). META-HEURISTIC ALGORITHMS FOR K-MEANS CLUSTERING: A REVIEW. PalArch's Journal of Archaeology of Egypt/Egyptology, 17(7), 12002-12020.

Zebari, D. A., Haron, H., Zeebaree, D. Q., & Zain, A. M. (2019, August). A Simultaneous Approach for Compression and Encryption Techniques Using Deoxyribonucleic Acid. In 2019 13th International Conference on Software, Knowledge, Information Management and Applications (SKIMA) (pp. 1-6). IEEE.

Eesa, A. S., Orman, Z., & Brifcani, A. M. A. (2015). A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems. Expert systems with applications, 42(5), 2670-2679.

Sadeeq, H., & Abdulazeez, A. M. (2018, October). Hardware implementation of firefly optimization algorithm using FPGAs. In 2018 International Conference on Advanced Science and Engineering (ICOASE) (pp. 30-35). IEEE.

Othman, G., & Zeebaree, D. Q. (2020). The Applications of Discrete Wavelet Transform in Image Processing: A Review. Journal of Soft Computing and Data Mining, 1(2), 31-43.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747.

Fleishman, G. M., & Thompson, P. M. (2017, April). Adaptive gradient descent optimization of initial momenta for geodesic shooting in diffeomorphisms. In 2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017) (pp. 868-872). IEEE.

Sadiq, S. S., Abdulazeez, A. M., & Haron, H. (2020). Solving multi-objective master production schedule problem using memetic algorithm. Indonesian Journal of Electrical Engineering and Computer Science, 18(2), 938-945.

Chandra, K., Meijer, E., Andow, S., Arroyo-Fang, E., Dea, I., George, J., ... & Yang, S. (2019). Gradient Descent: The Ultimate Optimizer. arXiv preprint arXiv:1909.13371.

Mohsin Abdulazeez, A., Hajy, D., Zeebaree, D., & Zebari, D. (2021). Robust watermarking scheme based LWT and SVD using artificial bee colony optimization. Indonesian Journal of Electrical Engineering and Computer Science, 21, 1218–1229.

"Gradient descent explained - Learn ARCore - Fundamentals of Google ARCore [Book]." https://www.oreilly.com/library/view/learn-arcore-/9781788830409/e24a657a-a5c6-4ff2-b9ea-9418a7a5d24c.xhtml (accessed Jan. 03, 2021).

Hardt, M., Recht, B., & Singer, Y. (2016, June). Train faster, generalize better: Stability of stochastic gradient descent. In International Conference on Machine Learning (pp. 1225-1234). PMLR.

Dogo, E. M., Afolabi, O. J., Nwulu, N. I., Twala, B., & Aigbavboa, C. O. (2018, December). A comparative analysis of gradient descent-based optimization algorithms on convolutional neural networks. In 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS) (pp. 92-99). IEEE.

Si, Z., Wen, S., & Dong, B. (2019). NOMA codebook optimization by batch gradient descent. IEEE Access, 7, 117274-117281.

Kaoudi, Z., Quiané-Ruiz, J. A., Thirumuruganathan, S., Chawla, S., & Agrawal, D. (2017, May). A cost-based optimizer for gradient descent optimization. In Proceedings of the 2017 ACM International Conference on Management of Data (pp. 977-992).

Saeed, J. N. (2020). A Survey of Ultrasonography Breast Cancer Image Segmentation Techniques. Infinite Study.

Cui, X., Zhang, W., Tüske, Z., & Picheny, M. (2018). Evolutionary stochastic gradient descent for optimization of deep neural networks. In Advances in neural information processing systems (pp. 6048-6058).

Wang, B., & Ye, Q. (2020). Stochastic Gradient Descent with Nonlinear Conjugate Gradient-Style Adaptive Momentum. arXiv preprint arXiv:2012.02188.

Guo, L., Li, M., Xu, S., & Yang, F. (2020, August). Application of Stochastic Gradient Descent Technique for Method of Moments. In 2020 IEEE International Conference on Computational Electromagnetics (ICCEM) (pp. 97-98). IEEE.

Zhang, J. (2019). Gradient descent based optimization algorithms for deep learning models training. arXiv preprint arXiv:1903.03614.

Yang, Z., Wang, C., Zhang, Z., & Li, J. (2019). Mini-batch algorithms with online step size. Knowledge-Based Systems, 165, 228-240.

Baydin, A. G., Cornish, R., Rubio, D. M., Schmidt, M., & Wood, F. (2017). Online learning rate adaptation with hypergradient descent. arXiv preprint arXiv:1703.04782.

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., & Han, J. (2019). On the variance of the adaptive learning rate and beyond. arXiv preprint arXiv:1908.03265.

Zhao, H., Liu, F., Zhang, H., & Liang, Z. (2019). Research on a learning rate with energy index in deep learning. Neural Networks, 110, 225-231.

Fei, Z., Wu, Z., Xiao, Y., Ma, J., & He, W. (2020). A new short-arc fitting method with high precision using Adam optimization algorithm. Optik, 164788.

Li, D., Chen, C., Lv, Q., Gu, H., Lu, T., Shang, L., ... & Chu, S. M. (2018, April). Adaerror: An adaptive learning rate method for matrix approximation-based collaborative filtering. In Proceedings of the 2018 World Wide Web Conference (pp. 741-751).

Du, S., Lee, J., Tian, Y., Singh, A., & Poczos, B. (2018, July). Gradient descent learns one-hidden-layer cnn: Don't be afraid of spurious local minima. In International Conference on Machine Learning (pp. 1339-1348). PMLR.

Mohammed, N. N., Cawthorne, M., & Abdulazeez, A. M. (2018). Detection of Genes Patterns with an Enhanced Partitioning-Based DBSCAN Algorithm. 9.

Eesa, A. S., Brifcani, A. M. A., & Orman, Z. (2014). A new tool for global optimization problems-cuttlefish algorithm. International Journal of Mathematical, Computational, Natural and Physical Engineering, 8(9), 1208-1211.

Zeebaree, D. Q., Abdulazeez, A. M., Hassan, O. M. S., Zebari, D. A., & Saeed, J. N. (2020). Hiding Image by Using Contourlet Transform.

De, S., Mukherjee, A., & Ullah, E. (2018). Convergence guarantees for RMSProp and ADAM in non-convex optimization and an empirical comparison to Nesterov acceleration. arXiv preprint arXiv:1807.06766.

Öztürk, M. M., Cankaya, I. A., & Ipekci, D. (2020). Optimizing echo state network through a novel fisher maximization based stochastic gradient descent. Neurocomputing, 415, 215-224.

Gylberth, R., Adnan, R., Yazid, S., & Basaruddin, T. (2017, October). Differentially private optimization algorithms for deep neural networks. In 2017 International Conference on Advanced Computer Science and Information Systems (ICACSIS) (pp. 387-394). IEEE.

Botev, A., Lever, G., & Barber, D. (2017, May). Nesterov's accelerated gradient and momentum as approximations to regularised update descent. In 2017 International Joint Conference on Neural Networks (IJCNN) (pp. 1899-1903). IEEE.

Najat, N., & Abdulazeez, A. M. (2017, November). Gene clustering with partition around mediods algorithm based on weighted and normalized Mahalanobis distance. In 2017 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS) (pp. 140-145). IEEE.

Sarigül, M., & Avci, M. (2018). Performance comparison of different momentum techniques on deep reinforcement learning. *Journal of Information and Telecommunication*, *2*(2), 205-216.

Gitman, I., Lang, H., Zhang, P., & Xiao, L. (2019). Understanding the role of momentum in stochastic gradient methods. In Advances in Neural Information Processing Systems (pp. 9633-9643).

Zeebaree, D. Q., Haron, H., Abdulazeez, A. M., & Zebari, D. A. (2019, April). Machine learning and Region Growing for Breast Cancer Segmentation. In 2019 International Conference on Advanced Science and Engineering (ICOASE) (pp. 88-93). IEEE.

"Momentum and Learning Rate Adaptation." https://www.willamette.edu/~gorr/classes/cs449/momrate.html (accessed Jan. 03, 2021).

Dozat, T. (2016). Incorporating nesterov momentum into adam.

Bargarai, F., Abdulazeez, A., Tiryaki, V., & Zeebaree, D. (2020). Management of Wireless Communication Systems Using Artificial Intelligence-Based Software Defined Radio.

Solanke, A. V. (2020). Intrusion Detection using Deep Learning Approach with Different Optimization. International Journal for Research in Applied Science and Engineering Technology, 8(5), 128–134. https://doi.org/10.22214/ijraset.2020.5022

Zhang, N., Lei, D., & Zhao, J. F. (2018). An improved Adagrad gradient descent optimization algorithm. In 2018 Chinese Automation Congress (CAC) (pp. 2359-2362). IEEE.

Bai, M., Liu, H., Chen, H., Gu, S., & Zhang, Z. (2019, October). An improved algorithm for radar adaptive beamforming based on machine learning. In Journal of Physics: Conference Series (Vol. 1325, No. 1, p. 012114). IOP Publishing.

Zeiler, M. D. (2012). Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701.

Nwankpa, C. E. (2020). Advances in optimisation algorithms and techniques for deep learning. Advances in Science, Technology and Engineering Systems Journal, 5(5), 563-577.

Qu, Z., Yuan, S., Chi, R., Chang, L., & Zhao, L. (2019). Genetic optimization method of pantograph and catenary comprehensive monitor status prediction model based on adadelta deep neural network. IEEE Access, 7, 23210-23221..

Zou, F., Shen, L., Jie, Z., Zhang, W., & Liu, W. (2019). A sufficient condition for convergences of adam and rmsprop. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 11127-11135).

Kirori, Z. (2019). Performance Analysis of Stochastic Gradient Descent-Based Algorithms for Time Series Sequence Modeling.

Mukkamala, M. C., & Hein, M. (2017). Variants of rmsprop and adagrad with logarithmic regret bounds. arXiv preprint arXiv:1706.05507.

Zebari, D. A., Zeebaree, D. Q., Saeed, J. N., Zebari, N. A., & Adel, A. Z. (2020). Image steganography based on swarm intelligence algorithms: A survey. people, 7(8), 9.

Keegan, B. (2018, October). Using First-Order Stochastic Based Optimizers in Solving Regression Models. In 2018 IEEE MIT Undergraduate Research Technology Conference (URTC) (pp. 1-4). IEEE.

Bock, S., Goppold, J., & Weiß, M. (2018). An improvement of the convergence proof of the ADAM-Optimizer. arXiv preprint arXiv:1804.10587.

Yaqub, M., Jinchao, F., Zia, M. S., Arshid, K., Jia, K., Rehman, Z. U., & Mehmood, A. (2020). State-of-the-Art CNN Optimizer for Brain Tumor Segmentation in Magnetic Resonance Images. Brain Sciences, 10(7), 427.

Fatima, N. (2020). Enhancing Performance of a Deep Neural Network: A Comparative Analysis of Optimization Algorithms. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal, 9(2), 79-90.

Vani, S., & Rao, T. M. (2019, April). An experimental approach towards the performance assessment of various optimizers on convolutional neural network. In 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI) (pp. 331-336). IEEE.

Yi, D., Ahn, J., & Ji, S. (2020). An Effective Optimization Method for Machine Learning Based on ADAM. Applied Sciences, 10(3), 1073.

Kemal, A. D. E. M., & KILIÇARSLAN, S. (2019, October). Performance Analysis of Optimization Algorithms on Stacked Autoencoder. In 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT) (pp. 1-4). IEEE.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

"ML Optimization Methods and Techniques." https://serokell.io/blog/ml-optimization (accessed Jan. 08, 2021).

Lydia, A., & Francis, S. (2019). Adagrad—An optimizer for stochastic gradient descent. Int. J. Inf. Comput. Sci., 6(5).

Endah, S. N., Widodo, A. P., Fariq, M. L., Nadianada, S. I., & Maulana, F. (2017, November). Beyond back-propagation learning for diabetic detection: Convergence comparison of gradient descent, momentum and Adaptive Learning Rate. In 2017 1st International Conference on Informatics and Computational Sciences (ICICoS) (pp. 189-194). IEEE.

Wibowo, A., Wiryawan, P. W., & Nuqoyati, N. I. (2019, May). Optimization of neural network for cancer microRNA biomarkers classification. In Journal of Physics: Conference Series (Vol. 1217, No. 1, p. 012124). IOP Publishing.

Tao, H., & Lu, X. (2018, July). On Comparing Six Optimization Algorithms for Network-based Wind Speed Forecasting. In 2018 37th Chinese Control Conference (CCC) (pp. 8843-8850). IEEE.

Lancewicki, T., & Kopru, S. (2020, May). Automatic and Simultaneous Adjustment of Learning Rate and Momentum for Stochastic Gradient-based Optimization Methods. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 3127-3131). IEEE.

Hapsari, D. P., Utoyo, I., & Purnami, S. W. (2020, October). Fractional Gradient Descent Optimizer for Linear Classifier Support Vector Machine. In 2020 Third International Conference on Vocational Education and Electrical Engineering (ICVEE) (pp. 1-5). IEEE.

Yu, Y., & Liu, F. (2019). Effective neural network training with a new weighting mechanism-based optimization algorithm. IEEE Access, 7, 72403-72410.

Kim, J., & Kim, H. (2017, February). An effective intrusion detection classifier using long short-term memory with gradient descent optimization. In 2017 International Conference on Platform Technology and Service (PlatCon) (pp. 1-6). IEEE.

Du, S., Lee, J., Li, H., Wang, L., & Zhai, X. (2019, May). Gradient descent finds global minima of deep neural networks. In International Conference on Machine Learning (pp. 1675-1685). PMLR.

Manogaran, G., & Lopez, D. (2018). Health data analytics using scalable logistic regression with stochastic gradient descent. International Journal of Advanced Intelligence Paradigms, 10(1-2), 118-132.

Zhang, J., Hu, F., Li, L., Xu, X., Yang, Z., & Chen, Y. (2019). An adaptive mechanism to achieve learning rate dynamically. Neural Computing and Applications, 31(10), 6685-6698.

Hong, H., Tsangaratos, P., Ilia, I., Loupasakis, C., & Wang, Y. (2020). Introducing a novel multi-layer perceptron network based on stochastic gradient descent optimized by a meta-heuristic algorithm for landslide susceptibility mapping. Science of the total environment, 742, 140549.

Sharma, A. (2018). Guided stochastic gradient descent algorithm for inconsistent datasets. Applied Soft Computing, 73, 1068-1080.

Ren, L. H., Ye, Z. F., & Zhao, Y. P. (2020). A modeling method for aero-engine by combining stochastic gradient descent with support vector regression. Aerospace Science and Technology, 99, 105775.

Pan, H., Niu, X., Li, R., Dou, Y., & Jiang, H. (2020). Annealed gradient descent for deep learning. Neurocomputing, 380, 201-211.