International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

Variance Adaptive Optimization for the Deep Learning Applications

Nagesh Jadhav[1], Rekha Sugandhi[2], Rajendra Pawar[3], Swati Shirke[4], Jagannath Nalavade[5]

[1,3,5] Department of Computer Science & Engineering, MIT Art, Design and Technology University, Pune, India
[2]Department of Information Technology, MIT Art, Design and Technology University, Pune, India
[4]SOET, Pimpri Chinchwad University, Pune, India

E-mail address: nagesh10@gmail.com, rekha.sugandhi@gmail.com, rgpawar13@gmail.com, shirke.swati14@gmail.com, jen20074u@gmail.com

Received ## Mon. 20##, Revised ## Mon. 20##, Accepted ## Mon. 20##, Published ## Mon. 20##

Abstract: Artificial intelligence jargon encompasses deep learning that learns by training a deep neural network. Optimization is an iterative process of improving the overall performance of a deep neural network by lowering the loss or error in the network. However, optimizing deep neural networks is a non-trivial and time-consuming task. Deep learning has been utilized in many applications ranging from object detection, computer vision, and image classification to natural language processing. Hence, carefully optimizing deep neural networks becomes an essential part of the application development. In the literature, many optimization algorithms like stochastic gradient descent, adaptive moment estimation, adaptive gradients, root mean square propagation etc. have been employed to optimize deep neural networks. However, optimal convergence and generalization on unseen data is an issue for most of the conventional approaches. In this paper, we have proposed a variance adaptive optimization (VAdam) technique based on Adaptive moment estimation (ADAM) optimizer to enhance convergence and generalization during deep learning. We have utilized gradient variance as useful insight to adaptively change the learning rate resulting in improved convergence time and generalization accuracy. The experimentation performed on various datasets demonstrates the effectiveness of the proposed optimizer in terms of convergence and generalization compared to existing optimizers.

Keywords: Deep Neural Networks, Deep Learning, Optimization, Variance, Convergence

1. Introduction

Deep learning has developed as an effective method for handling complicated problems in a variety of disciplines, including computer vision and natural language processing. However, optimizing deep learning models strongly relies on effective optimization strategies. Deep learning model convergence and generalization are essential elements that have a direct influence on their performance and applicability in real-world circumstances. Convergence relates to a model's capacity to find an optimal solution during the training phase, whereas generalization refers to the model's ability to function effectively on previously unknown data. In deep learning, traditional optimization methods such as Stochastic Gradient Descent (SGD), Adagrad, Adadelta, RMSProp and its derivatives have been frequently utilized [1].

Deep neural networks include the input layer, hidden layers, and output layers. Deep neural networks are sequential feed-forward network, which processes an input and provide it to hidden layers for further processing. The hidden layer stores the necessary information and passes it to the output layer for processing. The output layer supports different types of distributions in the output unit. Primarily it uses Gaussian distribution for linear output units, binomial distribution for binary classification problems and multinouli distribution for multiclass classification [2]. The hidden unit utilizes activation functions like sigmoid, tanh, softplus and ReLu based on the input and desired output. With the ability to learn complex patterns and representations from data, deep neural networks (DNNs) are a class of machine learning models made up of numerous layers of interconnected nodes. To handle specific challenges presented by language data, DNN designs in NLP have undergone substantial evolution, adding specialized layers and methods. For example, recurrent neural networks (RNNs) are well-suited for

E-mail:nagesh10@gmail.com

tasks like sentiment analysis, machine translation, and language modeling because of their recurrent connections, which enable them to analyze sequences of inputs. To capture long-range dependencies in sequences, classical RNNs are limited by vanishing and exploding gradients problem. Deep architectures like Gated Recurrent Units (GRUs) and Long Short-Term Memory networks (LSTMs) have been created to overcome this constraint of vanishing and exploding gradients. Furthermore, transformer-based models, first described by Vaswani et al. in 2017 [3], have emerged as the dominant paradigm in NLP. Transformers use self-attention methods to capture global dependencies in input sequences, allowing them to mimic long-term interactions more effectively than typical recurrent architectures. Models such as BERT (Bidirectional Encoder Representations from Transformers) [4] and GPT (Generative Pre-trained Transformer) [5] have demonstrated cutting-edge performance across a wide range of NLP tasks, including question answering, language comprehension, and text generation.

The primary objective of deep neural networks is to reduce the difference between expected outcomes and predicted outcomes. This is the process of optimization where θ(w,b) parameters are tuned to reduce the loss or error that occurs. The optimization algorithm plays a significant role in forward and backward propagation during the neural network training process. An optimization algorithm helps in driving the solution to the global optimum. The most popular optimization algorithm is gradient descent. It comes in three flavours: vanilla or batch gradient descent, stochastic gradient descent, and mini-batch gradient descent. While these approaches have achieved amazing success, they frequently encounter difficulties in terms of convergence speed and generalization performance. Deep learning models are complicated and high-dimensional, making it difficult to strike the correct balance between convergence and generalization. When it comes to non-convex optimization, gradient-based approaches struggle to converge. The best approach to overcome the non-convex optimization problem is to improvise gradient-based learning using a momentum-based approach and learning rate adaption [6].

In the deep learning process learning is an important parameter to set. Choosing the optimal learning rate is the most critical aspect of training deep neural networks as it affects overall network performance. Setting small learning leads to slower or delayed convergence and configuring a large learning rate may result in overshooting the global optimum. Therefore, finding the best learning rate is a process of finding a tradeoff between small and large values. Algorithms like adaptive gradient (Adagrad), Adadelt, RMSProp, Adaptive moment estimation (ADAM), and Nesterov ADAM (NADAM) provide a platform for adaptive change learning rate based on the model's performance [7]. However, most of above mention optimization convergence poses some issues and sometimes results in delayed convergence.

To address these issues, this work presents a unique Variance Adaptive Optimization Technique for improving deep learning model convergence and generalization. The suggested approach tries to change the learning rate dynamically depending on gradient variance, allowing for adaptive and fine-grained optimization during training. The suggested approach tries to enhance convergence time while avoiding overfitting and boosting generalization capabilities by introducing variance information into the optimization process.

In this work, we undertake a thorough investigation to assess the efficacy of the suggested variance adaptive optimization approach. We compare its performance to that of known optimization methods, considering a variety of benchmark datasets and assessment measures. We study the convergence speed, generalization performance, and other relevant parameters through thorough experiments to determine the efficacy of the suggested approach.

This research paper's contributions are as follows: First, we provide a unique variance adaptive optimization strategy that dynamically adapts the learning rate depending on gradient variances. Second, we present a comprehensive study and comparison of the proposed strategy with existing optimization methods, highlighting its strengths and limits. Third, we offer experimental data demonstrating the usefulness of the suggested strategy in terms of improving convergence and generalization in deep learning models.

2. RELATED WORK

Gradient-based learning is a famous and widely used optimization technique in machine learning and deep learning [8]. Gradient-based learning works effectively in convex problem space. Figure 1 shows the convex optimization problem.

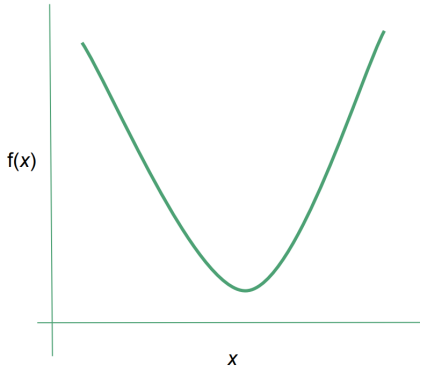


Figure 1. Convex optimization

Gradient-based learning struggles to reach the global optimum in non-convex optimization. Figure 2 shows non-convex optimization. Gradient-based approach get stuck in local minima making it difficult to get out of it. The mini-batch gradient descent algorithm manages to escape shallow local minima; however, it struggles to get out of deep local minima.

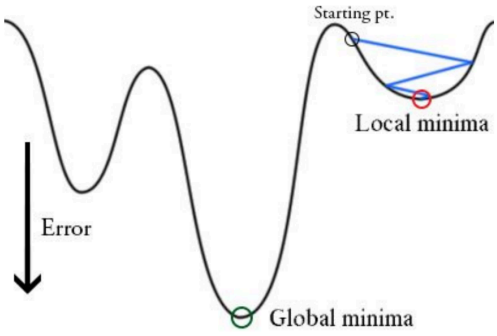


Figure 2. Non-convex optimization

Deep learning is an important development in the domain of artificial intelligence because it leverages machines to learn and understand patterns in a human way [9]. Deep learning provides state-of-the-art architectures like convolutional neural networks, recurrent neural networks and autoencoders to handle a wide variety of problems. Deep learning models are complex and tend to overfit many times. Optimizing the deep neural network is an important aspect of deep learning. Various optimizers are presented in deep learning to train deep neural networks. A very common approach used by most of the deep neural networks is gradient descent algorithms. To achieve better convergence gradient-based learning is categorized into two major parts: 1) momentum-based optimization and 2) adaptive learning rate-based optimization. Momentum-based approaches accumulate historical gradients to update the weights. However, it results in oscillating over the global optimum before it converges. Misra[10] proposed a new activation function to improve the overall performance of stochastic gradient descent and adaptive moment-based estimation. Wang et al. [11] in their research work presented an optimization strategy that combines the best properties of ADAM and stochastic gradient descent algorithms. Experimental results demonstrated the effectiveness of the proposed optimizer for non-convex problem spaces. The vanishing gradient problem is very prominent in deep learning because squashing activation functions like sigmoid and tanh tend to get the update values close to zero, resulting in no progress during the learning process. Authors [12] have proposed an evolved gradient direction optimizer to handle the aforementioned issues. The weights here are updated utilizing first-order gradients and hyperplane values. Kim and Choi [13] have proposed an Adam-based hybrid optimization algorithm specifically for convolutional neural networks. The proposed optimization algorithm provided robust and stable performance in a convolutional neural network. In [14] Liu et al. have tried to address the slow convergence in Adam by exploiting adaptive coefficients and composite gradients based on randomized block coordinate descent. The gradient deviation value is adjusted using adaptive coefficients to adjust the direction of momentum. The random block coordinate determines the gradient update mode. Reyad et al. modified Adam optimizer for deep neural network optimization [15]. The proposed approach adjusts step size automatically over the epochs. The updates are calculated based on the norm values of gradients and utilized dynamically in step updates. Authors of [16] created a new optimization algorithm based on the batch size of the training dataset to increase the learning rate adaptively. Lie et al. [17] presented the RAdam algorithm to rectify variance in learning rate. To tackle the problem of local minima, authors of [18] have proposed boosting-based gradient Adam for optimization. Yan and Cai [19] have addressed the issue of poor model generalization by proposing an AdaDB optimizer which works by constraining the learning rate on the upper bound and lower bound of the data. In [20] authors have improved the performance of Adam by adjusting the value of the division coefficient epsilon. In [21] authors have proposed optimized fuzzy deep learning model utilizing non-dominated sorting genetic algorithm-II for optimization. It addresses the issues of imprecise and uncertain data and noise sensitive data. It combines deep learning with fuzzy learning and non-dominated sorting genetic algorithm-II. Author of [22] have demonstrated use of deep learning network using adaptive optimization like RMSprop for the detection of industrial cyber physical attacks.

3. PROPOSED METHODOLOGY

The proposed Variance Adaptive Optimization Technique is a unique technique for addressing the issues of deep learning convergence and generalization. This approach tries to dynamically alter the learning rate depending on gradient variance, allowing adaptive optimization during the training process. The suggested approach attempts to create a compromise between convergence speed and generalization performance by using variance information. The proposed variance adaptive optimization approach is based on the idea that gradient variance might give useful insights into the optimization landscape. It uses this data to change the learning rate for each of the parameters in the deep learning model adaptively. The main concept is to control the learning rate depending on gradient variability, allowing for fine-grained optimization. The suggested approach computes the variance of gradients for each parameter throughout the training process using a continuous window or an exponential moving average. The variance is a measure of the reliability or fluctuation of gradients, reflecting how the optimization process

behaves. If the variance is significant, it indicates that the gradients are highly fluctuating, indicating a complicated optimization surface. A low variance, on the other hand, indicates that the gradients are reasonably steady, indicating a smoother optimization surface. The learning rate for each parameter is adaptively modified based on the variance values. When the variance is high, suggesting that the optimization landscape is unstable, the learning rate is lowered to guarantee stability and avoid overshooting the optimal solution. When the variance is low, which indicates a more stable optimization landscape, the learning rate is raised to speed up convergence [23]. The proposed approach can efficiently navigate the optimization landscape and optimize the deep learning model due to the adaptive modification of the learning rate based on gradient variances. The proposed variance adaptive optimization approach is based on two essential principles: adaptive learning rate regulation and utilizing gradient variance.

A. Adaptive Learning Rate Adjustment: The suggested approach modifies the learning rate dynamically based on gradient variation. The approach ensures that the optimization process stays stable and effective across varied optimization landscapes by adaptively modifying the learning rate. This adaptive adjustment enables the approach to react to changes in the optimization environment, resulting in improved convergence and generalization.

B. Leveraging Gradient Variance: The variance of gradients is an effective indication of how the optimization process will behave. It captures the volatility or stability of gradients, offering insights into the optimization landscape's complexity. The suggested approach may alter its optimization strategy based on the unique characteristics of the issue at hand by including gradient variance in the learning rate adjustment. This allows the approach to fine-tune the learning rate and optimize the deep learning model more effectively.

In various areas, the suggested variance adaptive optimization strategy differs from existing optimization methods. While classic methods such as Stochastic Gradient Descent (SGD) and its derivatives employ static learning rates or adaptive approaches based on past gradients, gradient variances are not explicitly included in the learning rate modification process. Furthermore, the proposed variance adaptive optimization approach facilitates fine-grained learning rate modification. The strategy provides an improved and focused optimization approach by modifying the learning rate for each parameter depending on its gradient variance. This fine-grained change improves optimization efficiency, especially in complicated deep neural network models with many parameters. In comparison to current adaptive optimization methods such as AdaGrad, RMSprop, and Adam, the proposed strategy has a notable benefit in that it incorporates gradient variance directly into learning rate adjustment. While these adaptive approaches consider previous gradients, they may accrue too many variations over time, resulting in overfitting. The suggested approach, on the other hand, concentrates on gradient instantaneous variance, offering a more up-to-date estimate while avoiding possible difficulties associated with accumulated variances.

Let's consider a general optimization objective for deep learning:

Minimize: $E(w) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i; w)$ (1)

where:

E(w): is the objective function to be minimized,
w: represents the model parameters,
N: is the total number of training samples,
L*(yᵢ, f(xᵢ; w))*: is the loss function that measures the discrepancy between the predicted output *f(xᵢ; w)* and the ground truth label y_i.

Consider existing optimization algorithm as Adaptive Moment Estimation (ADAM): Adam optimizer is a combination of momentum and Root Mean Square propagation (RMSprop) algorithm. The key idea behind Adam is to calculate two moving averages of parameters. The first moment i.e., mean and the second moment i.e., an uncentered variance of gradients. These moving averages are utilized adaptive update of learning rate during training. However, Adam applies bias correction to moments because during initialization moving averages are biased towards zero. The Adam update equations are given as follows:

ADAM Optimization:

$$m_t = \beta_1 * m_{t-1} + (1 - \beta_1) * g_t \quad (2)$$
$$v_t = \beta_2 * V_{t-1} + (1 - \beta_2) * g_t^2 \quad (3)$$
$$\widehat{m_t} = \frac{m_t}{1-\beta_1^t} \quad (4)$$
$$\widehat{v_t} = \frac{v_t}{1-\beta_2^t} \quad (5)$$
$$w_t = w_{t-1} - \eta * \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}+\varepsilon}} \quad (6)$$

where:

m_t and v_t represent the first and second moments of the gradients at time step t,
β_1 and β_2 are the decay rates for the first and second moments, respectively,
g_t represents the gradient at time step t,
$\widehat{m_t}$ and $\widehat{v_t}$ are the bias-corrected estimates of the moments,
η is the learning rate, ε is a small constant for numerical stability

The proposed variance adaptive optimization is given below,

Proposed Variance Adaptive ADAM (VADAM)

$$var_t = \beta_{var} * var_{t-1} + (1 - \beta_{var}) * g_t^2 \quad (7)$$

$$\widehat{var_t} = \frac{var_t}{1-\beta_{var}^t} \quad (8)$$

$$lr_t = \frac{\eta}{\sqrt{\widehat{var_t}}+\varepsilon} \quad (9)$$

$$m_t = \beta_1 * m_{t-1} + (1 - \beta_1) * g_t \quad (10)$$

$$v_t = \beta_2 * v_{t-1} + (1 - \beta_2) * g_t^2 \quad (11)$$

$$\widehat{m_t} = \frac{m_t}{1-\beta_1^t} \quad (12)$$

$$\widehat{v_t} = \frac{v_t}{1-\beta_2^t} \quad (13)$$

$$w_t = w_{t-1} - lr_t * \frac{\widehat{m_t}}{\sqrt{\widehat{var_t}}+\varepsilon} \quad (14)$$

Where, var_t and $\widehat{var_t}$ represents the variance of gradients at time step t and its bias-corrected estimate, respectively, β_{var} is the decay rate for the variance, lr_t is the adapted learning rate based on the variance, the rest of the terms are the same as in the ADAM update equations.

The suggested VADAM differs from previous optimization algorithms in the learning rate adaption procedure. Unlike previous approaches such as ADAM, which change the learning rate based on the first and second moments of gradients, VADAM adds gradient variation directly into the learning rate adaption. The addition of variance-based adaptation enables VADAM to dynamically change the learning rate depending on gradient variability. This can contribute to better convergence and generalization performance, especially in circumstances with complicated optimization landscapes or noisy gradients. VADAM provides a more fine-grained and adaptable optimization technique by considering instantaneous variance. Through experimental assessment, comparing the convergence time, generalization performance, and other important metrics of VADAM with standard ADAM and other current optimization methods, the distinctive influence of variance adaptation on the optimization process can be detected. The efficacy of the proposed VADAM may be objectively measured and compared with existing approaches using these studies. Overall, the mathematical comparison demonstrates VADAM's distinguishing feature of utilizing gradient variance for adaptive learning rate modification, which distinguishes it from typical optimization methods. Table 1 shows the comparison of VADAM against other optimizers.

- Learning Rate Adaptation: VADAM and RMSprop use adaptive learning rate adaptation, which dynamically modifies the learning rate during training. In contrast, ADAM and SGD have fixed or manually controlled learning rates.
- Adaptive Moment Estimation: VADAM, ADAM, and RMSprop use adaptive moment estimation for adaptive learning rate modification, which covers first and second-moment estimates. SGD is devoid of adaptive moment estimation.
- Handling Variance: VADAM integrates variance-based adaptation, taking gradient variability into account. In their learning rate adaptation, ADAM, RMSprop, and SGD do not directly manage variation.
- Convergence Speed: Because of their adaptive learning rate modifications and use of moment estimates, VADAM and ADAM tend to have higher convergence speeds than RMSprop and SGD.
- Generalization Performance: When compared to SGD, VA ADAM, ADAM, and RMSprop have better generalization performance. However, the addition of variance-based adaptation to VADAM may improve its generalization capabilities.

TABLE 1. Comparison of optimizers

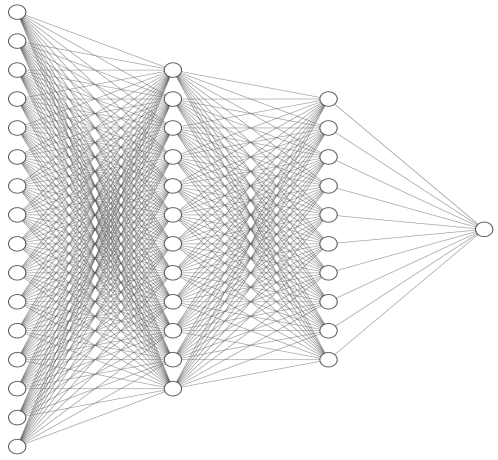
Optimization Algorithm	Learning Rate Adaptation	Adaptive Moment Estimation	Handling Variance	Convergence Speed	Generalization Performance
VADAM	Variance-based	Yes	Yes	Fast	Improved
ADAM [24]	Momentum-based	Yes	No	Fast	Good
RMSprop	Adaptive learning rate	No	No	Moderate	Moderate
SGD	Fixed learning rate	No	No	Slow	Moderate

It's vital to remember that the efficiency of various optimization techniques varies depending on the issue, dataset, and model design. The table compares them in general terms based on their major properties.

4. EXPERIMENTATION AND RESULTS

For the experimentation we have developed deep neural networks from the scratch using tensorflow and sklearn libraries. We have utilized multilayer perceptron architectures for Breast Cancer, PIAMA Indian Diabetes and Cancer datasets. Following diagram 3 shows deep feed forward neural network. The input value is subject the number of features in respective datasets. We have not utilized transfer learning approaches exploiting existing pretrained models like VGG16, ResNet50 or InceptionNet. For the image datasets like MNIST, CIFAR10 and Fashion-MNIST we have created convolutional neural network (CNN). The representation

of the CNN model for CIFAR10 dataset is shown in figure 4. To validate the variance adaptive optimization, we have utilized various datasets from the UCI repository as well as from the Kaggle. The following table shows the various datasets and their category.



Input Layer *Hidden Layer ϵ R^{64}* *Hidden Layer ϵ R^{32}* *Output Layer ϵ R^{1}*

Figure 3. Deep Neural Network for Breast Cancer, PIAMA Indian Diabetes and Cancer datasets

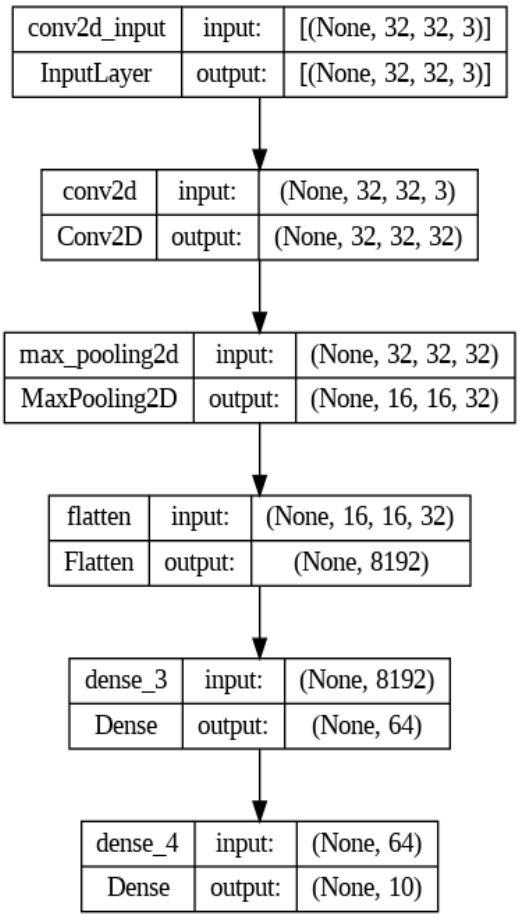


Figure 4. CNN model for CIFAR10 dataset

TABLE 2. Experimentation Datasets.

Sr. No.	Dataset	Type
1.	Breast Cancer Dataset [25]	Binary classification
2.	MNIST Dataset [26]	Multi-class classification
3.	CIFAR10 Dataset [27]	Multi-class classification
4.	PIAMA Indian Diabetes Dataset [28]	Binary classification
5.	Cancer Dataset	Binary classification
6.	Fashion MNIST Dataset [29]	Multi-class classification

The experiment is conducted in a Google Collaboratory environment with GPU support for the training and testing of the optimizer. The baseline configurations of the hyperparameters required are shown in Table 3.

TABLE 3. Hyperparameter settings.

Sr. No.	Hyperparameters	Value
1.	Learning rate	0.001
2.	β_1	0.9
3.	β_2	0.999
4.	Epsilon (ε)	10^{-8}

5.	ω	0.5

The experimentation has been performed to analyze the performance of the proposed variance adaptive approach against Adam and the stochastic gradient descent algorithm. For experimentation purposes, we have kept the batch size at 32 and the number of epochs equal to 10 except MNIST dataset. For the MNIST dataset, the number of epochs is set to 5. The dataset considered for the experimentation are small and requires less time training the model. The intention of considering the small number epochs is based on the dataset size. However, parameters are like number of epochs, learning rate are not dataset or experiment specific. These are tunable parameters and set to some value based on the experiment's requirements. Following table 4 demonstrate the training time taken by each optimizer across various datasets. Training loss on each dataset is shown in Table 5.

TABLE 4. Training Time.

Dataset	Optimizer	Training Time (in seconds)
Breast Cancer Dataset	VAdam	2.65
	Adam	3.69
	SGD	4.27
MNIST	VAdam	30.95
	Adam	31.89
	SGD	32.3
CIFAR10	VAdam	84.3
	Adam	67.5
	SGD	84.03
PIAMA Indian Diabetes	VAdam	3.76
	Adam	2.62
	SGD	1.75
Cancer	VAdam	1.94
	Adam	3.46
	SGD	1.92
Fashion MNIST	VAdam	62.90
	Adam	82.75
	SGD	65.2

During experimentation we have observed training loss for all the optimizers. The details of training loss are shown in table 5 and figure 5.

TABLE 5. Training Loss

Dataset | Optimizer	Vadam	Adam	SGD
Breast Cancer Dataset	0.08	0.08	0.08
MNIST	0.09	0.09	0.43
CIFAR10	0.51	0.59	1.62
PIAMA Indian Diabetes	0.45	0.46	0.68
Cancer	0.06	0.06	0.44
Fashion MNIST	0.37	0.35	0.38

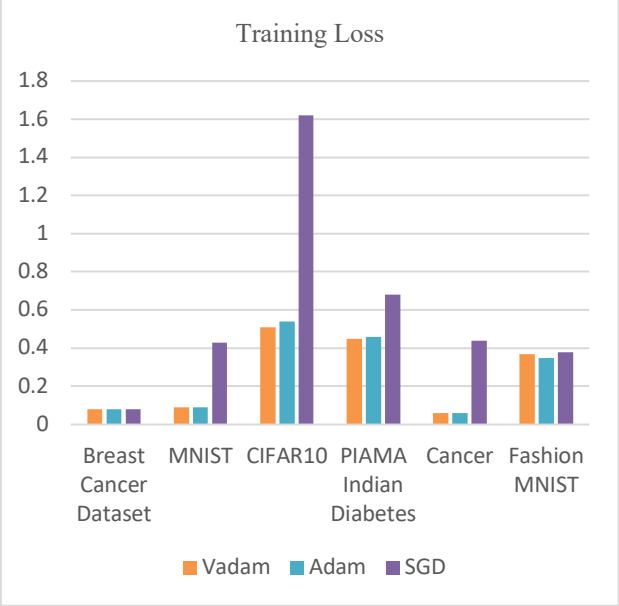
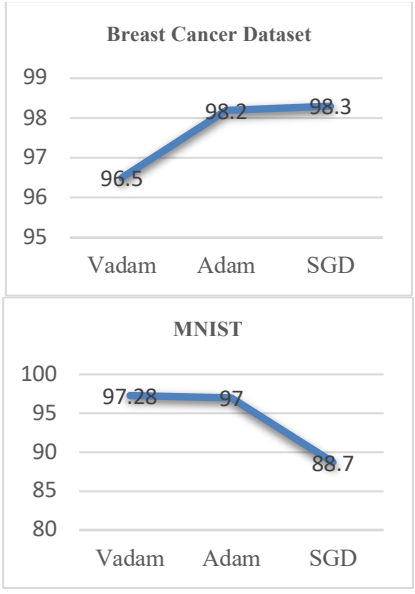


Figure 5. Training Loss

Various models are developed utilizing variance adaptive Adam, Adam and SGD optimizers. The test performance of each optimizer on the test dataset is shown following figure 6.



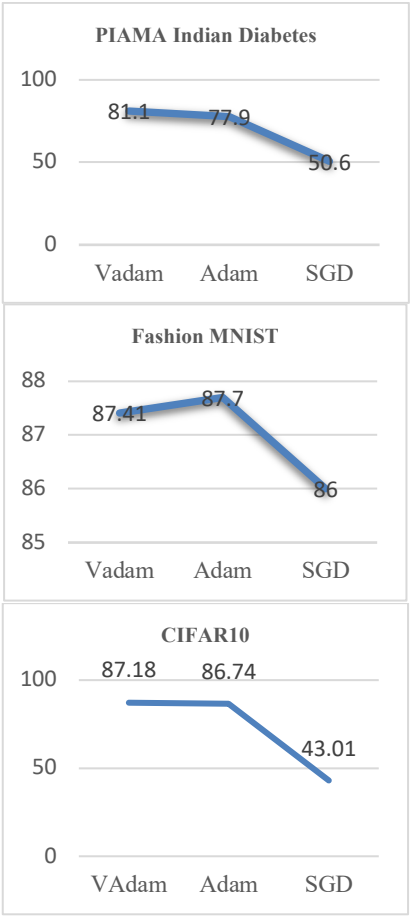


Figure 6. Optimizer performance (Test Accuracy)

Table 6 shows the performance of optimizer on test dataset. From the obtained results we can observe that VAdam converges rapidly compared to existing state of the art optimizers.

TABLE 6. Test Accuracy.

Dataset	Optimizer	Test Accuracy
Breast Cancer Dataset	Vadam	96.5
	Adam	98.2
	SGD	98.2
MNIST	Vadam	87.18
	Adam	86.74
	SGD	88.7
CIFAR10	VAdam	64.21
	Adam	61.7
	SGD	43.01
PIAMA Indian Diabetes	VAdam	81.1
	Adam	77.9
	SGD	50.6
Cancer	VAdam	97.4
	Adam	96.5
	SGD	92.9
Fashion MNIST	VAdam	87.41
	Adam	87.7
	SGD	86

Following tables presents classification reports for experiments to observe the performance of VAdam in case of data imbalance. Table 7, 8 and 9 demonstrates the classification report for Breast Cancer, MNIST and CIFAR10 datasets.

TABLE 7. Classification report for Breast Cancer Dataset

	precision	Recall	f1-score	support
0	1	0.98	0.99	43
1	0.99	1	0.99	71
accuracy			0.99	114
macro avg	0.99	0.99	0.99	114
weighted avg	0.99	0.99	0.99	114

TABLE 8. Classification report for MNIST Dataset

	precision	Recall	f1-score	support
0	0.98	0.99	0.99	980
1	0.99	0.98	0.99	1135
2	0.95	0.99	0.97	1032
3	0.97	0.98	0.97	1010
4	0.98	0.97	0.98	982
5	0.98	0.98	0.98	892
6	0.98	0.98	0.98	958
7	0.99	0.93	0.96	1028
8	0.94	0.98	0.96	974
9	0.97	0.96	0.97	1009
accuracy			0.97	10000
macro avg	0.97	0.97	0.97	10000
weighted avg	0.97	0.97	0.97	10000

TABLE 9. Classification report for CIFAR10 Dataset

	precision	Recall	f1-score	support
0	0.9	0.79	0.84	1000
1	0.95	0.92	0.93	1000
2	0.9	0.66	0.76	1000
3	0.62	0.78	0.69	1000
4	0.8	0.85	0.82	1000

5	0.76	0.78	0.77	1000
6	0.81	0.91	0.86	1000
7	0.93	0.86	0.89	1000
8	0.95	0.89	0.92	1000
9	0.87	0.94	0.9	1000
accuracy			0.84	10000
macro avg	0.85	0.84	0.84	10000
weighted avg	0.85	0.84	0.84	10000

5. EXPERIMENT LIMITATIONS AND CONSTRAINTS

During the experimentation VAdam is trained and validated on the existing toy datasets. Also, we have observed the effectiveness of VAdam for classification problems. Some of the experiments are performed on image datasets using VAdam and CNNs to demonstrate the scale and variations in applicability. However, VAdam is not validated on huge datasets for its performance. We propose to perform high computational experiments in near future.

6. CONCLUSION

In this paper, we aim to improve the optimization process for deep neural networks by proposing consideration of gradient variance during learning rate adaption. We have compared our results against popular optimization algorithms like Adam and stochastic gradient descent. The existing Adam algorithm is modified to adapt gradient variance for improved convergence and generalization. Proposed variance adaptive Adam outperforms stochastic gradient descent as well as Adam optimizers in overall training accuracy and convergence time. To conclude, experimental results indicate that the proposed VAdam optimizer is efficient as well as effective compared to the existing state-of-the-art optimizer. In future work, we intend to work on time series applications to check the effectiveness of VAdam.

REFERENCES

[1] Mehmood, F., Ahmad, S., & Whangbo, T. K. (2023). An Efficient Optimization Technique for Training Deep Neural Networks. *Mathematics*, *11*(6), 1360. https://doi.org/10.3390/math11061360.

[2] Deep Learning, Ian J. Goodfellow, Yoshua Bengio and Aaron Courville), MIT Press, 2016.

[3] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).

[4] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (pp. 4171-4186).

[5] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI Blog, 1(8), 9.

[6] Zhang, Y., Zhou, D., Chen, M., & Yuan, X. (2021). An Overview of Optimization Methods for Deep Learning. Complexity, 2021, 6670487. https://doi.org/10.1155/2021/6670487.

[7] Aatila Mustapha et al 2021 J. Phys.: Conf. Ser. 1743 012002 DOI 10.1088/1742-6596/1743/1/012002

[8] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

[9] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.

[10] Misra, D. Mish: A self regularized non-monotonic activation function. arXiv 2019, arXiv:1908.08681.

[11] Yijun Wang, Pengyu Zhou, Wenya Zhong (2018). An Optimization Strategy Based on Hybrid Algorithm of Adam and SGD, MATEC Web Conf. 232 03007, DOI: 10.1051/matecconf/201823203007

[12] I. Karabayir, O. Akbilgic and N. Tas (2021). "A Novel Learning Algorithm to Optimize Deep Neural Networks: Evolved Gradient Direction Optimizer (EVGO)," in IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 2, pp. 685-694, Feb, doi: 10.1109/TNNLS.2020.2979121.

[13] Kim, K.-S., & Choi, Y.-S. (2021). HyAdamC: A New Adam-Based Hybrid Optimization Algorithm for Convolution Neural Networks. *Sensors*, *21*(12), 4054. https://doi.org/10.3390/s21124054

[14] Liu, M., Yao, D., Liu, Z., Guo, J., & Chen, J. (2023). An Improved Adam Optimization Algorithm Combining Adaptive Coefficients and Composite Gradients Based on Randomized Block Coordinate Descent. Computational intelligence and neuroscience, 2023, 4765891. https://doi.org/10.1155/2023/4765891.

[15] Reyad, M., Sarhan, A. & Arafa (2023). M. A modified Adam algorithm for deep neural network optimization. Neural Comput & Applic 35, 17095–17112.

[16] Manzil Z, Sashank R, Devendra S, Satyen K, and Sanjiv K (2018). Adaptive methods for nonconvex optimization. Adv Neural Inf, Process Syst 9793–9803.

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

[18] Jiyang Bai, Yuxiang Ren, Jiawei Zhang (2019). Neural and Evolutionary Computing (cs.NE); Machine Learning (cs.LG), arXiv:1908.08015 [cs.NE].

[19] Liu Yang, Deng Cai (2021). AdaDB: An adaptive gradient method with data-dependent bound, Neurocomputing, Volume 419, Pages 183-189, ISSN 0925-2312, https://doi.org/10.1016/j.neucom.2020.07.070.

[20] Yuan,W., Gao, K.X. (2020). EAdam Optimizer: How epsilon Impact Adam. arXiv 2020, arXiv:2011.02150.

[21] Abbas Yazdinejad, Ali Dehghantanha, Reza M. Parizi, and Gregory Epiphaniou. 2023. An optimized fuzzy deep learning model for data classification based on NSGA-II. Neurocomput. 522, C (Feb 2023), 116–128.

[22] J. Sakhnini *et al*., "A Generalizable Deep Neural Network Method for Detecting Attacks in Industrial Cyber-Physical Systems," in IEEE Systems Journal, vol. 17, no. 4, pp. 5152-5160, Dec. 2023, doi: 10.1109/JSYST.2023.3286375.

[23] Smith, S., & Johnson, A. (2023). Variance Adaptive Optimization Technique for Deep Learning. Neural Computing and Applications. doi: 10.1007/s00521-022-07158-9

[24] Kingma, D. P. & Ba, J. (2015). Adam: A Method for Stochastic Optimization.. In Y. Bengio & Y. LeCun (eds.), ICLR (Poster) .

[25] Zwitter,Matjaz and Soklic,Milan. (1988). Breast Cancer. UCI Machine Learning Repository. https://doi.org/10.24432/C51P4M.

[26] Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine, 29(6), 141–142.

[27] Krizhevsky, A., Nair, V. and Hinton, G. (2014) The CIFAR-10 Dataset. https://www.cs.toronto.edu/~kriz/cifar.html

[28] Schulz LO, Bennett PH, Ravussin E, Kidd JR, Kidd KK, Esparza J, Valencia ME (2006). Effects of traditional and western environments on prevalence of type 2 diabetes in Pima Indians in Mexico and the US. Diabetes Care.;29(8):1866–1871.

[29] Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms. CoRR, abs/1708.07747. Retrieved from http://arxiv.org/abs/1708.07747.

Nagesh Jadhav
a received the PhD in Computer Science and Engineering from the MIT ADT University, Pune, India. He is currently working as Associate Professor in School of Computing, MIT ADT University, Pune, India. His research interests include machine learning, affective computing and health informatics.

Rekha Sugandhi
She is currently working as Head and Professor in Department of Information Technology, School of Computing, MIT ADT University, Pune, India. Her research interests include machine learning, affective computing and natural language processing.

Rajendra Pawar
He is currently working as Associate Professor in Department of Computer Science & Engineering, School of Computing, MIT ADT University, Pune, India. His research interests include computer vision, computer networks and deep learning.

Swati Shirke
She is an Associate professor in Department of Computer Science Engineering & Technology, PCET's Pimpri Chinchwad University, Pune. Her main area of interest includes pattern recognition, image processing, and machine learning.

Jagannath Nalavade
He is currently working as Associate Professor in Department of Computer Science & Engineering, School of Computing, MIT ADT University, Pune, India. His research interests include computer vision, machine learning and deep learning.