ADAM-Add: Enhancing ADAM with Adaptive Decay Rates

Lemeng Dai
Department of Computer Science
University of Toronto
lemeng.dai@mail.utoronto.ca

Allison Tsz Kwan Lau
Department of Computer Science
University of Toronto
allison.lau@mail.utoronto.ca

Wenrui Wu
Department of Computer Science
University of Toronto
wenrui.wu@mail.utoronto.ca

Abstract

Optimizers are algorithms that find a set of model parameters which minimizes the error of prediction. As an attempt to improve on current optimizers, we propose Adam-Add, a modification to the Adam optimizer and preliminary investigation on it. Adam-Add updates its decay rates each timestep according to current gradient magnitude. This modification aims to address Adam's sensitivity to decay rates. Four experiments were performed to test Adam-Add's performance against Adam and variants of Adam. The experiments include logistic regression, MLP and CNN being trained on MNIST, CIFAR and IMDb datasets. The results show that in high curvature loss landscapes, Adam-Add has a better performance but in low curvature, Adam-Add is slow in convergence. We also found that the optimizer does not converge in complicated landscapes. We believe that it would require more rigorous analysis and further modifications to fix the convergence issue.

1 Introduction

Optimizers play a crucial role in training machine learning models by adjusting the parameters of the model in order to minimize the loss function. A good optimizer should find the best local optimum in the loss function landscape efficiently. Minimizing the loss between the prediction and ground truth and finding the model that fits the data, but at the same time it should not overfit the data.

Some optimizers that have been proposed before include 1) Gradient Descent which is the most basic optimizer, 2) SGD (Stochastic Gradient Descent) which is an extension of gradient descent that computes the gradient using mini-batch rather than the entire dataset, making it more computationally efficient, 3) AdaGrad (Duchi et al. [2011]), which achieves significantly better performance than SGD by scaling the gradients by square roots of some averaging of the squared of past gradients, and to tackle the problem of deteriorating performance in dense gradient and non-convex situations in AdaGrad, 4) RMSprop (Root Mean Square Propagation) (Tieleman [2012]), another adaptive learning rate optimization algorithm that maintains a moving average of squared gradients to normalize the learning rates.

Adam (Kingma and Ba [2017]) is an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments. It combines the benefits of RMSprop and Adagrad, and introduces a bias-correction step. It is one of the most widely used optimizer, with the update rule is given by

$$\begin{aligned}
m_t &= \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot \nabla_\theta f(\theta_t) \\
v_t &= \beta_2 \cdot v_{t-1} + (1-\beta_2) \cdot (\nabla_\theta f(\theta_t))^2 \\
\hat{m}_t &= \frac{m_t}{1-\beta_1^t} \\
\hat{v}_t &= \frac{v_t}{1-\beta_2^t} \\
\theta_{t+1} &\leftarrow \theta_t - \frac{\alpha}{\sqrt{\hat{v}_t}+\varepsilon} \cdot \hat{m}_t
\end{aligned}$$

where m_t and v_t are first and second moment estimates of gradients, β_1 and β_2 are the decay rates for moment estimates, and ε is a small constant to avoid division by zero.

In this project, we identify a potential shortcoming of the Adam optimizer; Adam's performance can be sensitive to the momentum decay rates (β_1 and β_2). If this momentum decay rate can be adjusted according to the current time step in the training process, Adam's performance will be more stable. In this project, we propose a potential improvement to Adam to make the momentum decay rate adaptive to current gradient to further enhance Adam's convergence.

2 Prior work

Much work has been conducted to mitigate several issues of Adam or enhance its performance. AdamW in Loshchilov and Hutter [2019] improves regularization in Adam by decoupling the weight decay from the gradient-based update. NAdam in Dozat provides preliminary evidence that incorporating Nesterov's accelerated gradient improves the speed of convergence and the quality of the learned models. RAdam in Liu et al. [2021] introduces a term to rectify the variance of the adaptive learning rate. AMSGrad in Reddi et al. [2019] fixes a convergence issue of Adam. In this work, we incorporate fixes by AMSGrad and include new parameter updates. We aim to further enhance the performance of Adam.

3 Algorithm

Algorithm 1 is our modified Adam optimizer.

In section 3.1, we will discuss the modifications made to the Adam optimizer. In section 4, we discuss the convergence of the modified algorithm. In section 5, we will empirically investigate the performance of our proposed method with experiments using logistic regression, multi-layer neural network, and convolutional neural network.

3.1 Proposed modifications: Updating β_1 and β_2

To address the shortcoming of Adam's sensitivity to decay rates, we propose to have adaptive decay rates that adjust to gradient magnitude, that is, at time step t, β_1 and β_2 would have the following update.

$$\begin{aligned}
\beta_{1,t} &= \beta_1 \cdot e^{-\lambda_1 \cdot |\nabla_\theta f(\theta_t)|_\infty} \\
\beta_{2,t} &= \beta_2 \cdot e^{-\lambda_2 \cdot |\nabla_\theta f(\theta_t)|_\infty}
\end{aligned}$$

where λ_1, λ_2 are the sensitivity parameters in $\mathbb{R}^+$, and the norm on the gradient being infinity norm. Since the norm is non-negative, the exponential term is bounded in $(0, 1]$. The intuition behind the equations is that we decrease β_1 and β_2 when gradients are large and increase it when gradients are small, ensuring that the optimizer is less influenced by past gradients when current gradients are significant. We tested empirically using different orders of norm, and the performance is similar, but we would recommend using infinity norm.

Additionally, Reddi et al. [2019] pointed out convergence issues with the original Adam optimizer, along with the proposal of AMSGrad optimizer, with $\hat{v}$ updated to be the max of $\hat{v}_{t-1}$ and v_t. We have adapted this change into our algorithm.

Algorithm 1 Our proposed algorithm for stochastic optimization. The algorithm is based on Adam, with all operations involving vectors to be pointwise.

```
Require: $\alpha$: Initial stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $\lambda_1, \lambda_2 \in \mathbb{R}^+$: Sensitivity of $\beta$ adjustment
Require: $f(\theta)$: Stochastic objective function with parameters $\theta$
Require: $\theta_0$: Initial parameter vector
  $m_0 \leftarrow 0$                                   ▷ Initialize 1st moment vector
  $v_0 \leftarrow 0$                                   ▷ Initialize 2nd moment vector
  $t \leftarrow 0$                                     ▷ Initialize timestep
  while $\theta_t$ not converged do
      $t \leftarrow t + 1$
      $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$      ▷ Get gradients w.r.t. stochastic objective at timestep $t$
      $\beta_{1,t} \leftarrow \beta_1 \cdot e^{-\lambda_1 \cdot |g_t|_\infty}$
      $\beta_{2,t} \leftarrow \beta_2 \cdot e^{-\lambda_2 \cdot |g_t|_\infty}$
      $m_t \leftarrow \beta_{1,t} \cdot m_{t-1} + (1 - \beta_{1,t}) \cdot g_t$      ▷ Update biased first moment estimate
      $v_t \leftarrow \beta_{2,t} \cdot v_{t-1} + (1 - \beta_{2,t}) \cdot g_t^2$    ▷ Update biased second raw moment estimate
      $\hat{m}_t \leftarrow m_t / (1 - \beta_{1,t}^t)$      ▷ Compute bias-corrected first moment estimate
      $\hat{v}_t \leftarrow \max\{\hat{v}_{t-1}, v_t\}$
      $\hat{v}_t \leftarrow \hat{v}_t / (1 - \beta_2^t)$    ▷ Compute bias-corrected second raw moment estimate
      $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon)$
  end while
  return $\theta_t$                                    ▷ Updated parameters
```

4 Convergence discussion

Kingma and Ba [2017] provides a theoretical convergence guarantee for Adam. However, this was pointed out to be flawed by Reddi et al. [2019], leading to the proposal of the AMSGrad optimizer, as mentioned in the previous section. There are also multiple convergence analysis of Adam (Wang et al. [2022], Zou et al. [2019], Défossez et al. [2022], De et al. [2018]). Although our optimizer is very similar to AMSGrad in Reddi et al. [2019], there are situations (such as in section 5.3) in which our optimizer diverges. In our work, we only provide preliminary evidence our optimizer performs well in some cases. A more rigorous analysis would be needed to find out the constraints on β.

Adam was pointed out to not converge in certain cases, even with simple convex objective functions (Reddi et al. [2019]). Works mentioned above try to propose modifications to mitigate this issue. From convergence analyses of those works, most make use of the assumption that $\beta_{1,t} = \beta_1 \lambda^{t-1}$ for some $\lambda \in [0, 1]$ for any time step t in order to conclude that the proposed Adam's variant, such as AMSGrad, ensures a regret of $\mathcal{O}(\sqrt{T})$, or at least, is bounded. This is possible for those algorithms since they do not update β, so λ could simply take the value of 1 and the relevant theorem would hold. However, in our case, although $\beta_{1,t}$ is kept at a value between β_1 and 0, it is not non-increasing. Besides this, since β_1, β_2 are free to change based on the gradient, another assumption in the proofs, which is that $\beta_1/\sqrt{\beta_2} < 1$, might not be satisfied in all time steps. This is a probable reason why our algorithm does not converge in some cases.

In light of this, in section 5.3, we have tried to add the update $\lambda_t \leftarrow \lambda_t/\sqrt{t}$ to force the convergence of the optimizer. In this case, our optimizer does converge but at a lower rate than the original Adam. To compensate for the lower rate, we also attempted to reset the time step t after a certain time step. More discussion can be found in section 5.3.

4.1 Convergence path

To evaluate characteristics of optimization algorithms, we tested and compared the convergence path of our optimizer to that of Adam and AMSGrad, the two closest algorithms to ours. We used three common artificial landscapes, including Rosenbrock's function, Himmelblau's function and Easom's function and visualize the convergence path of the optimizers on these landscapes. In all three landscapes, our optimizer converges. We have used the same hyper-parameters as specified in section 5. The results are shown in Figures 1 to 4.

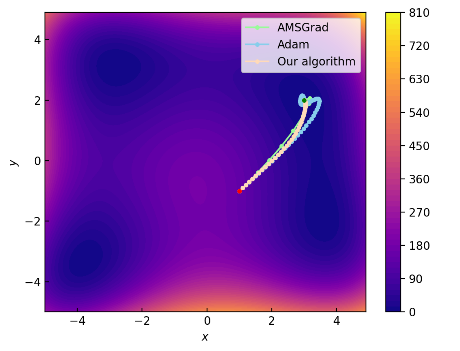

Figure 1: Path of convergence on Himmelblau's function which has several minima. The starting point, marked in red, is at $(1, -1)$. The green dot indicates the nearest true global minimum $(3, 2)$ of the function. Our optimizer does not overshoot like Adam and AMSGrad.

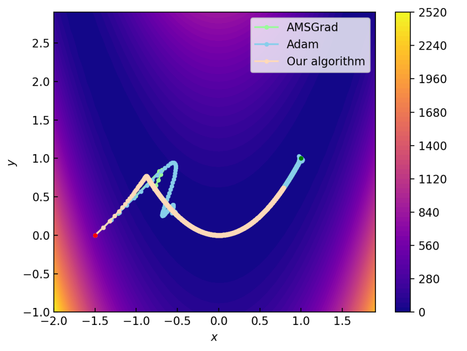

Figure 2: Path of convergence on Rosenbrock's function which has a ravine landscape. The true minimum marked in green is at $(1, 1)$. Our algorithm converges slower than Adam and AMSGrad. However, it has a stabler update in initial iterations with no oscillation.

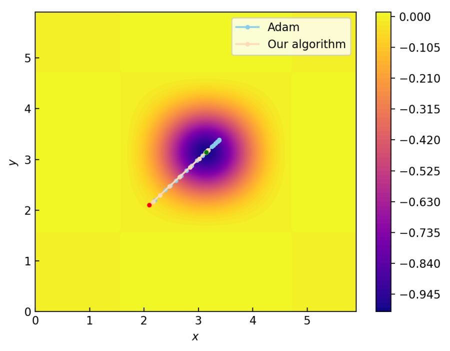

Figure 3: Path of convergence on Easom's function. Adam and our algorithm are compared. Our optimizer overshoots less than Adam. The green dot at (π, π) indicates the true minimum. The red dot indicates the starting point.

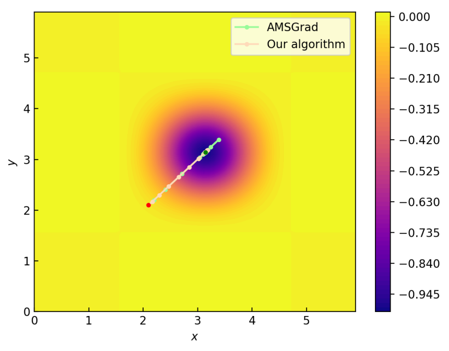

Figure 4: Path of convergence on Easom's function. AMSGrad and our algorithm are compared. Our optimizer overshoots less than Adam. The green dot at (π, π) indicates the true minimum. The red dot indicates the starting point.

In both landscapes, our algorithm chooses a more direct path to the convergence point and is less likely to overshoot the minimum as in the case for Adam and AMSGrad when using the exact same hyper parameters, as in Figure 1, 3 and 4. This shows the effect of the β update in stabilizing the update. In particular, we find that the β_2 update is more significant in suppressing momentum, as compared to β_1. However, as in Figure 2, in regions with small gradient, the convergence is very slow since the optimizer might be too cautious and strict in making big updates.

5 Experiments

Besides artificial landscapes, we perform experiments on real world dataset and models. First, we perform a grid search to find the best parameters λ_1, λ_2 and report the best hyper-parameter setting. An initial search was done over $[0.5, 1]$. It was found that the loss was the lowest at around 0.8, hence the search was performed on finer intervals in the range $[0.8, 0.9]$. It was found that the best value was $\lambda_1 = 0.8, \lambda_2 = 0.8$. We used the recommended value of $\beta_1 = 0.9, \beta_2 = 0.999$. We used a learning rate of 0.005. Experiments in sections 5.1, 5.2, and 5.3 were all included in Adam's paper as well (Kingma and Ba [2017]).

5.1 Experiment: Logistic regression using MNIST and IMDb review dataset

We compare the performance of our modified Adam with that of Adam, and two other variants of it, including NAdam and RAdam on simple logistic regression using the MNIST dataset (LeCun and Cortes [2005]). As mentioned by Kingma and Ba [2017], logistic regression has a well-studied convex objective, making it suitable for comparison of different optimizers without worrying about local minimum issues. We have found that for our modified Adam, the performance, in terms of both convergence rate and loss value, is faster without the decaying learning rate $\alpha_t = \alpha_1/\sqrt{t}$, where α_1 is the initial learning rate, which was used in Adam in Kingma and Ba [2017] in practice. We believe that this is because combining with our β update, the decay on the learning rate is too aggressive. We used 0.7, 0.1, 0.2 split for training, validation, and test sets. The network used is one single linear layer followed by a softmax layer. For all the optimizers, we have used a batch size of 128 and the full set of data. According to Figure 5, our algorithm has similar convergence as NAdam and Adam. Our algorithm converges slightly faster and is more stable than Adam, with smaller fluctuation as it approaches convergence. To account for the randomness in each training, for each optimizer, we trained the model 5 times and averaged the loss at each iteration. The result is shown in Figure 6.

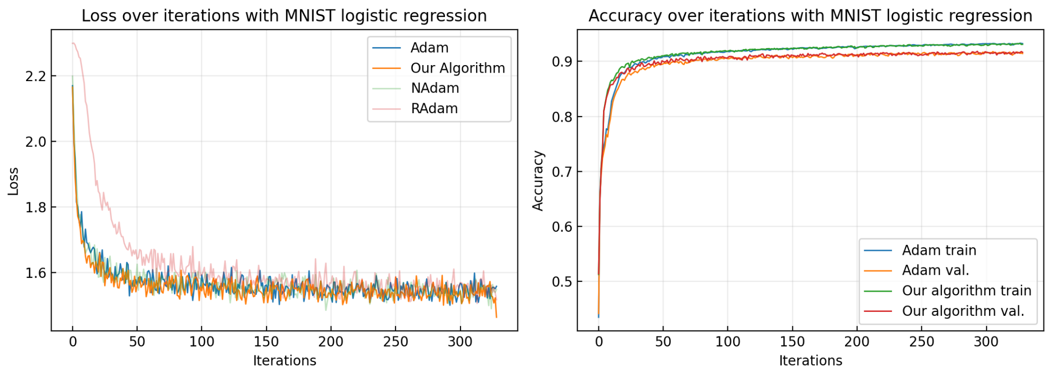

Figure 5: Logistic regression training negative log likelihood on MNIST images. For loss over iterations, two variants of Adam are shown.

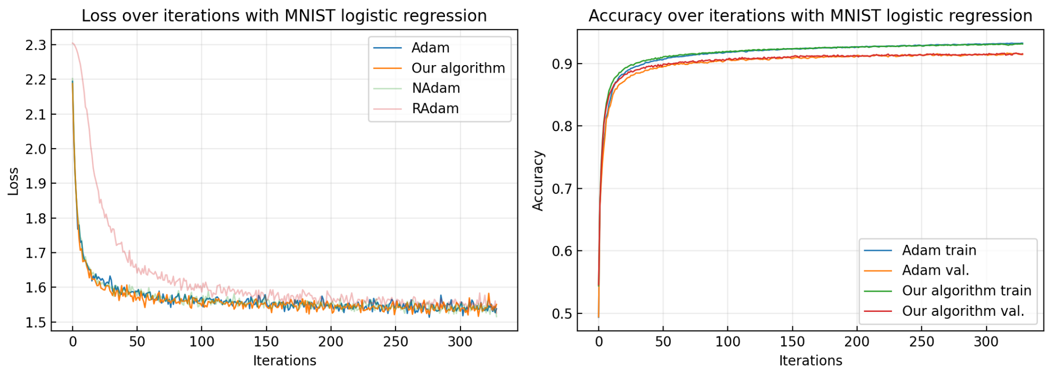

Figure 6: Averaged Logistic regression training negative log likelihood on MNIST images. Adam and our algorithm are compared.

From Figure 6, our algorithm converged to approximately the same loss and the same training and validation accuracy as Adam and Nadam. Our algorithm converged at a slightly faster rate than Adam.

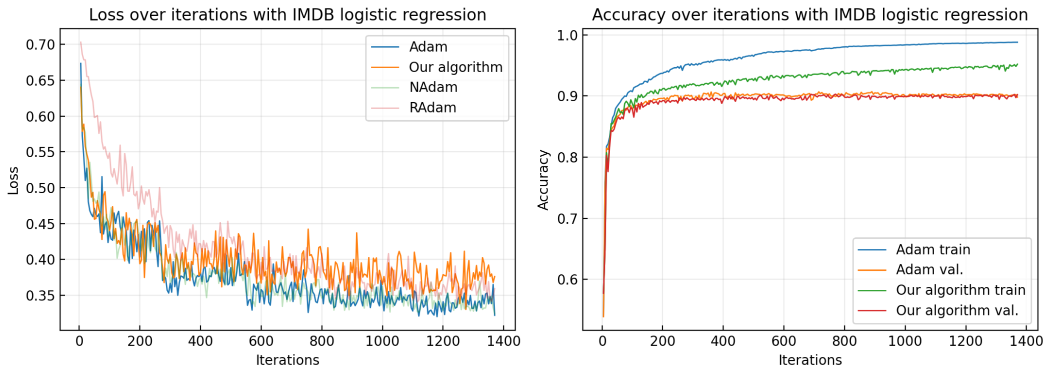

Figure 7: Logistic regression training negative log likelihood on IMDb reviews dataset. Our algorithm is compared with the original Adam and two variants of Adam.

Aside from the MNIST dataset, we have used the IMDB movie review dataset from (Maas et al. [2011]). Similar to the MNIST dataset, we used the full set of data and batch size of 128. According to Figure 7, our algorithm has slightly higher loss than Adam and NAdam, and the convergence is not as stable as Adam. We believe that it might be that the IMDb dataset, have different characteristics compared to the MNIST dataset. It may have more complex and noisy data patterns, making it more challenging for the optimizer to converge efficiently.

5.2 Experiment: Multi-layer neural networks using MNIST

Multi-layer neural networks have non-convex objective functions. We use a model that is consistent with Kingma and Ba [2017] and with previous publications in this area. Our multi-layer neural network consists of two fully connected hidden layers, each with 1000 units followed by ReLU activation and dropout with 0.5 probability. The batch size used is 128, and all the hyper-parameters, including the learning rate, $\lambda_1, \lambda_2, \beta_1, \beta_2$ are the same as in the logistic regression experiment in the previous section. We also study different optimizers using the standard deterministic cross-entropy objective function with L2 regularization on the parameters. Figure 8 shows the performance of our algorithm compared to Adam on the MNIST dataset.

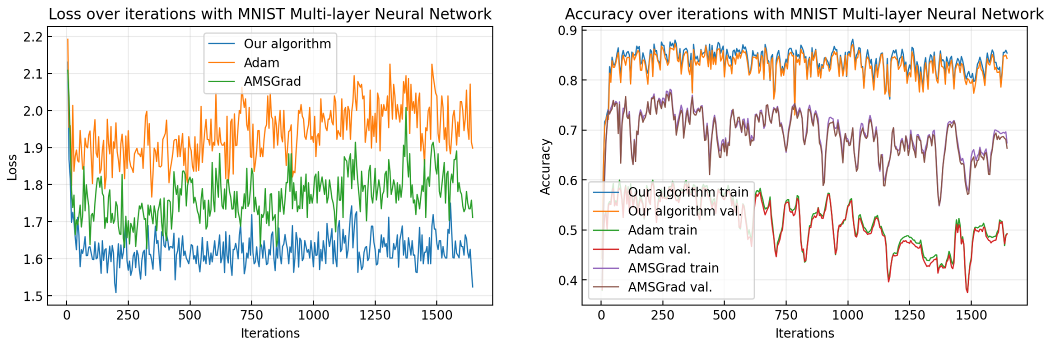

Figure 8: Training of multilayer neural networks on MNIST images, various optimizers are shown.

According to Figure 8, our algorithm outperforms both the original Adam and AMSGrad on loss, training accuracy and validation accuracy. Our algorithm has a smaller loss than Adam after the first initial iterations. We have included AMSGrad in the comparison to investigate the significance of the β-update. Results show that our algorithm does perform better than AMSGrad with this model and dataset. It can be seen that the update improves the adaptiveness of the algorithm, which allows it to converge at a lower loss.

5.3 Experiment: Convolutional neural network using CIFAR-10

Weight sharing in CNNs results in vastly different gradients in different layers. We have found that our optimizer does not perform well in CNNs. We tested the performance of our optimizer on the CIFAR-10 dataset (Krizhevsky et al.) with a CNN architecture consisting of three convolutional layers with 5×5 convolution filters followed by a 3×3 max pooling layer with a stride of 2, and one fully connected layer of 1000 ReLUs, consistent with Kingma and Ba [2017]. The batch size used is 128. We have used the full set of data with a 0.8, 0.1, 0.1 split for training, validation, and test set. It is found that applying our modified optimizer, the loss does not converge. The results are shown in Figure 9. We believe that this might due to our modification making the update of v_t, m_t more

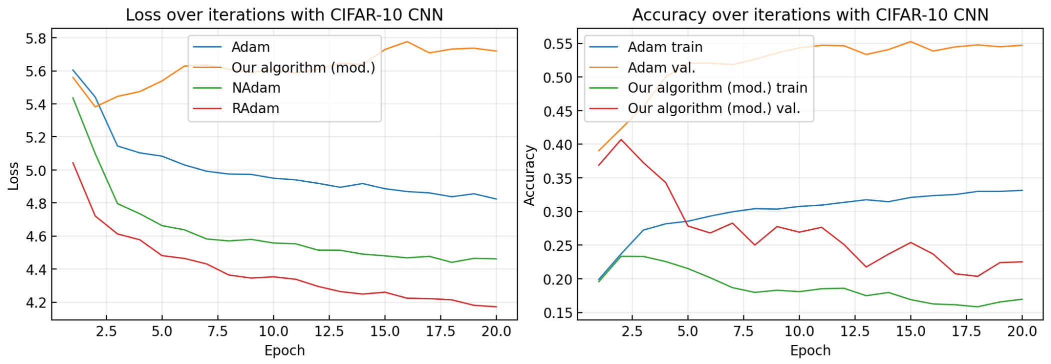

Figure 9: Training of convolutional neural networks on CIFAR-10 dataset, loss and accuracy using Adam and our optimizer are shown.

extreme and both of them vanishing to 0 after some iterations thus it won't converge. We propose a solution to that as mentioned in section 4, and the result after this modification is shown in Figure 10.

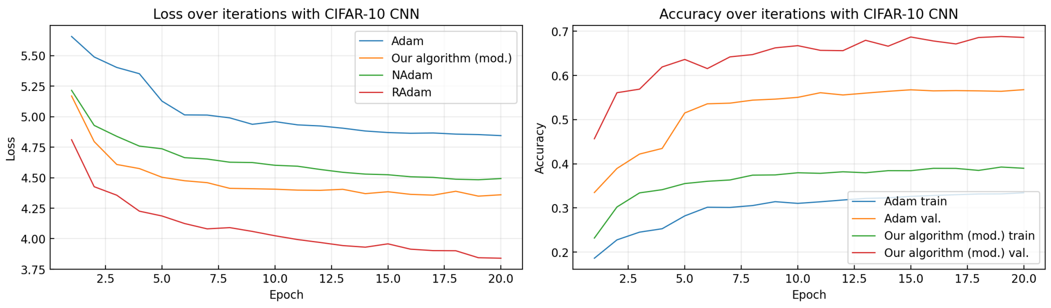

Figure 10: Training of convolutional neural networks on CIFAR-10 dataset, Adam and our algorithm are shown.

According to Figure 10, our algorithm performs better than Adam after the modification. The modification was to decrease the value of λ with respect to the time step and reset the time step. By doing this, we avoid the value of our momentum vanishing while allowing it to go faster and allow our loss to converge. And if our loss can converge successfully, we obtain a better result than Adam.

6 Conclusion

We proposed a modification to the Adam optimizer. In addition to the maximum of past squared gradient adpated by AMSGrad, we added an update to the decay rates β_1 and β_2. Following this modifications, we visualized the convergence on several artificial landscapes, and tested our optimizer empirically on logistic regression, multi-layer neural network, and CNNs. Initially, we believe that adding an adaptive decay rate will stabilize the update, which will improve the speed of convergence.

However, in practice, as shown in our experiments, the result was not always positive. In addition, in cases where our optimizer converges, although the optimizer was stable, it is sometimes too cautious and thus result in slow convergence. Nonetheless, we do find in some experiments that our additional update to β improves Adam in both the rate of convergence and the loss at convergence.

Our optimizer would require more rigorous mathematical analysis to investigate the constraints with which it would be able to surpass existing Adam variants. There are also other possible extensions for the optimizer, for example integrating the Nesterov momentum such as in NAdam. More experiments with more complex models and diverse datasets could be performed to assess Adam-Add's optimization ability in different scenarios.

7 Description of Individual Contribution

Allison Lau and Wenrui Wu came up with the idea of how to modify the Adam optimizer. Allison Lau wrote the algorithm for our new optimizer and convergence discussion. Wenrui Wu wrote the code for the new optimizer. All three of us participate in the experiment: Allison Lau did the multi-layer neural network experiment, Wenrui Wu did the CNN experiment and Lemeng Dai did the logistic regression experiment. All of us participate in writing the report equally.

8 Supplementary Material

Our code for producing the plots found in the report can be found in the GitHub repository https://github.com/LemengDai/CSC413-Project.

References

Soham De, Anirbit Mukherjee, and Enayat Ullah. Convergence guarantees for rmsprop and adam in non-convex optimization and an empirical comparison to nesterov acceleration, 2018.

Timothy Dozat. Incorporating Nesterov Momentum into Adam. In *Proceedings of the 4th International Conference on Learning Representations*, pages 1–4.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011. URL `http://jmlr.org/papers/v12/duchi11a.html`.

Alexandre Défossez, Léon Bottou, Francis Bach, and Nicolas Usunier. A simple convergence proof of adam and adagrad, 2022.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

Yann LeCun and Corinna Cortes. The mnist database of handwritten digits. 2005. URL `https://api.semanticscholar.org/CorpusID:60282629`.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL `https://aclanthology.org/P11-1015`.

Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond, 2019.

T. Tieleman. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude, 2012. URL `https://cir.nii.ac.jp/crid/1370017282431050757`.

Bohan Wang, Yushun Zhang, Huishuai Zhang, Qi Meng, Zhi-Ming Ma, Tie-Yan Liu, and Wei Chen. Provable adaptivity in adam, 2022.

Fangyu Zou, Li Shen, Zequn Jie, Weizhong Zhang, and Wei Liu. A sufficient condition for convergences of adam and rmsprop, 2019.