Math 303 Project Proposal

Adaptive Moment Estimation

PREPARED BY

Galal Mohamed

Anas Ahmed

Mohamed Hassan Ismail

Mohamed Ehab Yousr

i Ibrahem Ali

Introduction

In deep learning, the optimization algorithms represent the major ways of training neural networks efficiently and effectively. Among these optimizers, one of the most popular is Adam, which stands for Adaptive Moment Estimation; it is well-known due to its adaptation capabilities regarding the learning rate for each parameter, which usually leads to fast convergence and better performance. However, Adam itself has many variants, each of which has been designed to handle specific challenges. The examples are AdamW, decoupling weight decay for better regularization, Adamax is more robust to noisy gradients, while AMSGrad avoids slow convergence due to ensuring consistent optimization. This report discusses the behavior and performance of Adam and its variations. It compares the strengths and weaknesses of these variants over different tasks and datasets.

1. Problem Definition

The problem addressed in this proposal revolves around the limitations and challenges associated with the Adaptive Moment Estimation (Adam) optimizer and its variations.
Adam has emerged immensely popular as a practical optimization technique for stochastic objectives due to its several advantages, but actually, there are some weaknesses in its generalization, stability, and convergence. These problems stand in large-scale machine learning problems where one of the most important features is optimization robustness.

One of the major difficulties that the first model using Adam has is the dependence on biased estimators of the first and second moments of the gradients. Bias in the moment estimators could therefore be detrimental to the convergence, (e.g., in convex optimization), where Adam is not guaranteed to converge to critical points. Also, it has been found as a big disadvantage the inordinate ability of controlling a decreasing learning rate schedule and that is when dealing with noisy or sparse gradients.

Mathematical Formulation:

-The optimization problem is defined as minimizing an objective function.

where:

- set of parameters whose values are to be determined after optimization.
- a measure of how far off the model predicted output from the real output.
- Input data sampled from the distribution.

Objective Function:

$$J(\theta) = E(x, y) \sim D[\ell(f(\theta; x), y)]$$

Where:

θ: The parameters of the model to be optimized.

$(x, y) \sim D$: The input x and the corresponding output label y, which comes from distribution D.

$\ell(f(\theta; x), y)$: a loss function defined on the output f(θ;x) of the model with parameters θ and the true label y.

$(\theta; x)$: The model's predicted output for input x.

Regardless of its algorithms, this formulation has limitations. For instance, the exponential moving averages of the gradients (0 and 1) ensure a non-negative learning rate which leads to non-convergence in some cases. In this context, there are AMSGrad-type algorithms that use the modified update rule mentioned above, which in turn reduces the probabilities of ever-increasing number of instances by relatively controlling the rate of learning.

This project aims to propose and solve some of the challenges of analyzing Adam and its variants e.g. Adamax, AMSGrad, and AdamW in terms of performance on stability, convergence, and adaptability. The attention will be also directed towards developing new optimization algorithms that will improve their strength and extend their usability on various machine learning tasks.

---

2. Literature Review

The first version of the Adaptive Moment Estimation optimizer [1] was introduced as a solution to the challenges found in traditional stochastic optimization methods. It was referred to as a lightweight optimizing algorithm designed to overcome the limitations of AdaGrad and RMSProp by the ability to work with non-stationary objectives. The main innovation of Adam lies in computing individual learning rates for each parameter, making it particularly effective for extensive machine learning tasks requiring fast convergence with minimal memory requirements which was the main highlighted feature in this algorithm and being introduced as a solution for broader and faster ML algorithms optimizing approach

However, this initial formulation was not without limitations. The ADAM had issues in terms of its proposed convergence behavior. For example, as highlighted by Dereich and Jentzen [2], the basic version of Adam can fail to converge to the critical points of strongly convex objective functions under certain conditions. This occurs due to the algorithm's reliance on biased estimates of the gradient's first and second moments, which, while effective in many scenarios, can lead to suboptimal or divergent behavior in others especially in dealing with large datasets [3]. These issues started a family of sub-algorithms of the ADAM to refine and improve the issues addressed in each version.

In order to overcome these drawbacks, [3] investigated Adam's non-convergence characteristics, paying particular attention to the issues that arise when exponential moving averages are used. The problem of the anticipated gradient's dependence on its history is identified as critical when they present an instance of a convex optimization problem in which Adam's convergence cannot be achieved. In addition to highlighting the fundamental shortcomings of Adam's original formulation, this research led to the creation of AMSGrad, a variation that takes into account long-term memory of previous gradients. Because it strives to maintain a non-increasing learning rate during its use, AMSGrad stops the learning rate from fading. Additionally, AMSGrad ensures that all issues pertaining to divergence are adequately addressed while concurrently preserving all of Adam’s benefits for computation and user ease. This refinement exemplifies the iterative process of algorithmic enhancement within the broader context of adaptive optimization methods.

Based on the main ideas of ADAM, [4] integrated it into Beetle Antennae Search (BAS), which resulted in an improved algorithm called BAS-ADAM. This paper used a hybrid approach that integrated the global search features of BAS with the adaptive learning rate feature of ADAM to improve the speed and accuracy in convergence. BAS-ADAM showed perfect performance in solving complex optimization problems with balancing the trade-offs between exploration and exploitation. The experiments showed the algorithm’s adaptation capability to the change in landscapes of the problems, making it a robust choice for problems that require efficient optimization. Additionally, the research indicated the potential of hybrid algorithms in advancing optimization techniques by focusing on improving some particular components.

In optical waveguide mode solvers, Adam optimizer has proven to be quite helpful. A key application involves its incorporation into the Pseudospectral Frequency-Domain (PSFD) framework to improve its accuracy and stability. By using Adam, researchers can quickly find the penalty coefficients required to solve eigenvalue problems, such the two-dimensional Helmholtz equation, and impose boundary conditions. Adam provides better robustness to local minima, faster convergence rates, and more computational efficiency compared to metaheuristic techniques like the Gorilla Troops Optimizer (GTO) and more conventional techniques like Strong Boundary Methods (SBM). This has been especially noticeable in intricate waveguide structures with many subdomains or sharp corners. As a result, Adam has emerged as a crucial instrument for improving the accuracy and functionality of optical component analysis and design [5].

Adam achieves faster convergence in early epochs as it has high efficiency and adoptable learning rates, but it can face overfitting issues sometimes, but Stochastic Gradient Descent (SGD) with momentum offers smoother convergence over time and better generalization. Alternatives like AdaBelief, which improves generalization by modifying step sizes according to gradients, and Padam, which incorporates partial adaptiveness to balance generalization and convergence speed, solve Adam's problems. Padam has shown that it has the ability to improve performance in some scenarios by resolving the tiny learning rate issue of the adaptive approaches [6].

3. Methodology

How does it work?

Adam is an adaptive optimization technique that tracks two moments of the gradient automatically to adjust the learning rate of each parameter.

1. Compute the gradient

   At each step t, we compute the gradient of the loss function with respect to the model's parameters. We'll call it g_t

   $$g_t = \nabla_0 L(\theta_t)$$

   L is our loss function

   θ_t is the parameter at each step

2. Update the first moment

   The first moment is the average of the gradients. Adam calculates the exponential decay of the moving average of the gradients

   $$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\, g_t$$

β_1 is the decay for the first moment

m_t is the biased first moment

3. Update the second moment

   The second moment is the average of the squared gradient. We track the exponential decay of the average of the squared gradients.

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

   β_2 is the exponential decay for the second moment

   v_t is the biased second moment (variance)

4. Correct the bias

   Adam corrects these biases by dividing them by factors that depend on how many updates we have done.

$$\hat{m} = \frac{m_t}{1-\beta_1^t}$$

$$\hat{v} = \frac{v_t}{1-\beta_2^t}$$

5. Update the parameter

   In the previous steps, we corrected 2 moments. We will use these moments to update each parameter θ_t

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

   α is our base learning rate

ϵ is just a small number to prevent division by 0

Advantages

1. Faster start: Adam converges much quicker than many methods in the start of training.
2. Memory-friendly: Adam doesn’t require more memory than standard methods
3. Robust to local minima: It can easily escape tricky regions in the search space, making it useful in complex optimization problems and sure in real life.

Disadvantages

1. Potential Overfitting: Because it converges so quickly, it overfits easily if not tuned or regularized.
2. Learning Rate Decay: Some versions can get stuck if the learning rate drops so quickly.

Variations of Adam

There have been proposed several variations of Adam to improve it.

Variation	Description	Benefits
AdamW	Decouples weight decay from the gradient update.	More stable training and better generalization.
Adamax	Uses the infinity norm instead of the L2 norm for scaling the learning rate.	More robust to outliers and noisy gradients.
AMSGrad	Uses a "long-term memory" of past gradients.	Addresses a potential convergence issue in Adam.

These variations offer different trade-offs. For example, AdamW addresses the issue of weight decay with the optimization process, while Adamax provides increased robustness to noisy gradients[7]. Different Adam variants also address vanishing and exploding gradient problems differently[8].

Empirical Evaluation

To evaluate the performance of Adam and its variations (AdamW, Adamax, AMSGrad) standard benchmark datasets like MNIST, CIFAR-10, and ImageNet are used. They are specifically interested in various aspects that can measure the efficiency and effectiveness of optimizers in deep learning. Convolutional neural networks (CNNs) are utilized in the modeling as these are widely used for the task of image classification.

The experiments are designed with a uniform setup including a selected machine-specific learning rate and batch size, and for a fixed number of epochs aiming at convergence. Each optimizer is evaluated against the following key measures.

Convergence Speed: Measured by the rate of loss reduction.

Accuracy: Measured by the model result in the testing dataset after the model training is completed.

Loss: Measured by the stability of the training as well as cognizance of the optimization.

This experiment is implemented using TensorFlow, which provides support for all variations of Adam. Results are visualised using Matplotlib, and Numpy is used for numerical calculations.

Visual Results

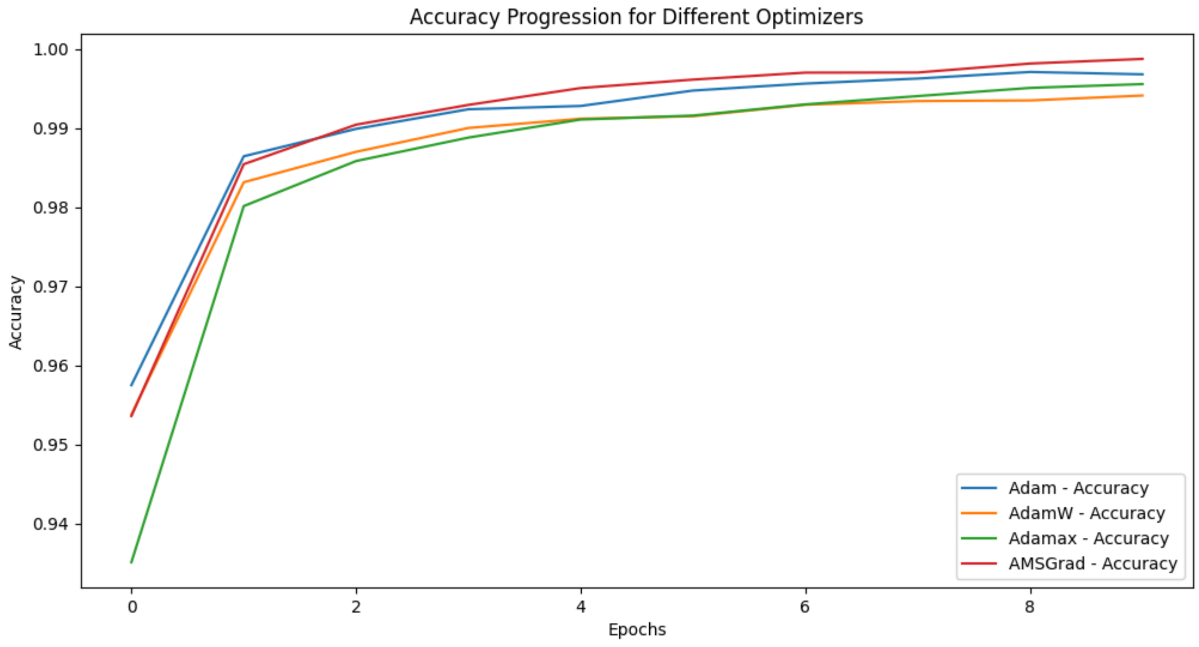

Figure 1: Accuracy progression for Adam, AdamW, Adamax, and AMSGrad during training.

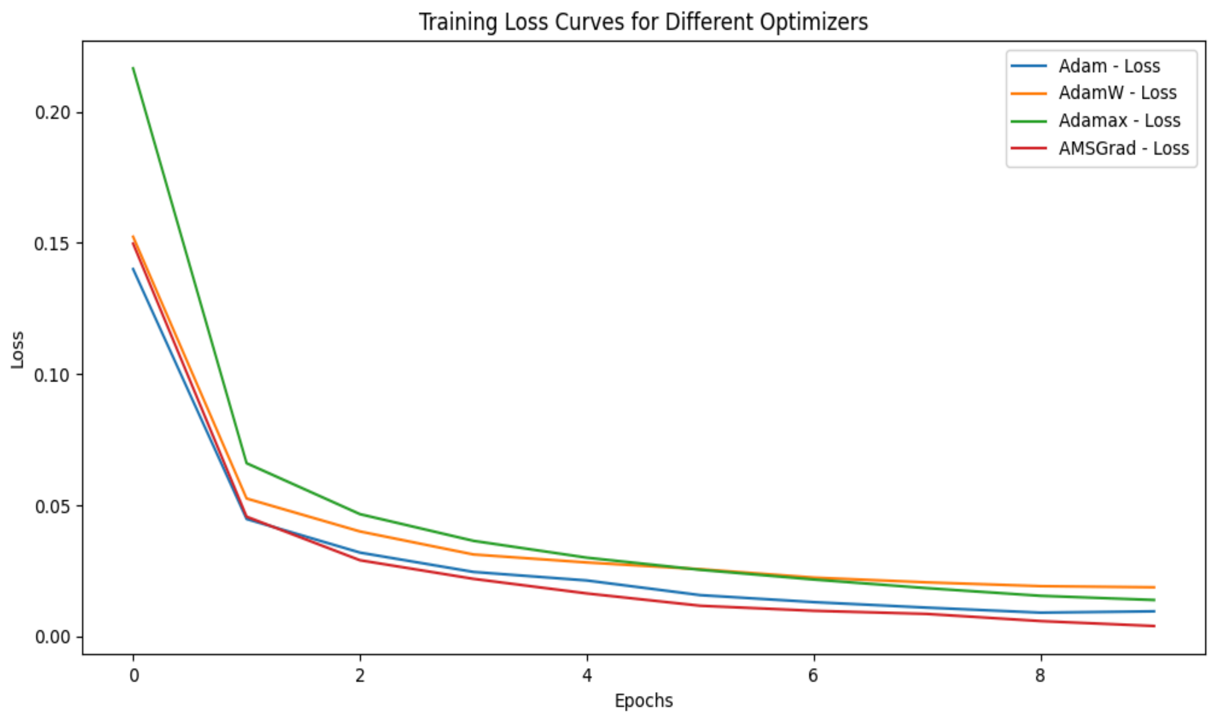

Figure 2: Training loss curves for Adam, AdamW, Adamax, and AMSGrad.

Suitability for Different Tasks

Every variant of Adam, including AdamW, Adamax, and AMSGrad, has properties that make its application more effective for a particular task. It is important to understand such differences, as these allow one to choose the best optimizer for a particular job.

AdamW: This is suitable for problems where weight decay needs to be decoupled from the gradient update to enhance the model's generalization and stability. This is mostly recommended in tasks, such as image classification and object detection scope, where regulation is necessary.

Adamax: This is suitable for natural language processing tasks due to its tolerance properties toward noisy and large gradients. Being infinity norm-based, it is less sensitive to outliers, hence suitable for sparse datasets or datasets with unusual gradient distributions.

AMSGrad: This is suitable for reinforcement learning and training deep networks that require consistent convergence to the same solution. AMSGrad makes sure that moving averages used in adam do not get the optimizer stuck in a suboptimal solution; it alleviates convergence issues, hence reliable for complex tasks demanding stable and consistent convergence.

4. References (IEEE)

[1] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” *arXiv (Cornell University)*, Jan. 2014, doi: 10.48550/arxiv.1412.6980.

[2] S. Dereich and A. Jentzen, “Convergence rates for the Adam optimizer,” *arXiv (Cornell University)*, Jul. 2024, doi: 10.48550/arxiv.2407.21078.

[3] S. J. Reddi, S. Kale, and S. Kumar, “On the Convergence of Adam and Beyond,” *arXiv (Cornell University)*, Jan. 2019, doi: 10.48550/arxiv.1904.09237.

[4] A. H. Khan, X. Cao, S. Li, V. N. Katsikis, and L. Liao, “BAS-ADAM: an ADAM based approach to improve the performance of beetle antennae search optimizer,” *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 2, pp. 461–471, Mar. 2020, doi: 10.1109/jas.2020.1003048.

[5] P.-J. Chiang, “Adaptive penalty method with an Adam optimizer for enhanced convergence in optical waveguide mode solvers,” *Optics Express*, vol. 31, no. 17, p. 28065, Jul. 2023, doi: 10.1364/oe.495855.

[6] J. Chen, D. Zhou, Y. Tang, Z. Yang, Y. Cao, and Q. Gu, “Closing the generalization gap of adaptive gradient methods in training deep neural networks,” *arXiv.org*, Jun. 18, 2018. https://arxiv.org/abs/1806.06763

[7] “AdamW — PyTorch 2.5 documentation.”
https://pytorch.org/docs/stable/generated/torch.optim.AdamW.html

[8] Astitwa, “Comparative analysis of Adam Optimizer and its variants: Adamax, RMSProp, and Adagrad — Addressing vanishing and exploding gradient problems,” *Medium*, Nov. 25, 2024. [Online]. Available:
https://medium.com/@astitwa15108824/comparative-analysis-of-adam-optimizer-and-its-variants-adamax-rmsprop-and-adagrad-addressing-02889adc8735