

GALGOTIAS UNIVERSITY
SCHOOL OF COMPUTING SCIENCE & ENGINEERING

BCSE4881- (B.Tech-CAPSTONE PROJECT- 2)


Undertaking Form for Project MTE-2024-25

I, Keshav Kumar & Jayant Sikarwar UG Student of B.Tech-CSE, Admission
No: 21SCSE1180121, 21SCSE1180159 hereby declare that,

1. As per the guidelines I need to have 1 publication for taking my Final ETE exam.
2. The research paper is verified by the Supervisor before publishing the article.
3. The Project report has less than 15% plagiarism.
4. The publication details are as follows,

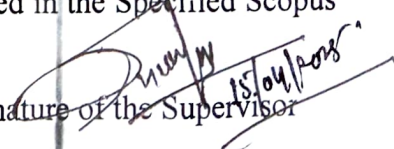
A. My Research article is published in (Journal Name, ISSN No, Vol, Page No)/Accepted
in
.....
.....
.....
.....

B. My Research article submitted in (Waiting for the
Acceptance) International Conference on Information
Systems & Computer Networks (ISCON 2025)
.....
.....
.....
.....


Signature of the Student

As per the university norms, **I Undertake the total responsibility of the student related to Publication of 1 research article in Scopus conference/journal.** Also I have gone through the Presentation, Dissertation report of the research work. I **approve/ Reject** the work and he/she can take **Final Review cum Viva voce** as per the guidelines.

Note: If the Student got Accepted their research Article in Scopus Conference, the supervisor should ensure that it has been presented in the conference and published in the Specified Scopus conference/journal


Signature of the Supervisor

HYPERPARAMETER OPTIMIZATION USING VARIOUS OPTIMIZERS

Keshav Kumar¹

Department of Computer Science and Technology
Galgotias University
Greater Noida, UP
Keshav.kr2825@gmail.com

Jayant Sikarwar²

Department of Computer Science and Technology
Galgotias University
Greater Noida, UP
jayantsikarwar205.js@gmail.com

Sugan Patel³

Department of Computer Science and Technology
Galgotias University
Greater Noida, UP
Sugan.patel@gmail.com

ABSTRACT—Hyperparameter optimization plays a crucial role in enhancing the performance of machine learning models by fine-tuning parameters. In this study, we evaluate the performance of four widely-used optimization algorithms—Adam, RMSprop, SGD, and Adagrad—on the MNIST handwritten digits classification task using a Convolutional Neural Network (CNN). Grid search was employed to systematically explore combinations of hyperparameters, including learning rate, batch size, number of dense layers, and neurons per layer.

The results highlight the impact of each optimizer's characteristics on training efficiency, accuracy, and loss. Adam demonstrated the best overall performance with the highest accuracy and fastest convergence, while RMSprop closely followed. In contrast, SGD and Adagrad required longer training times but showed competitive performance under specific configurations. This comparative analysis provides insights into the optimal choice of optimizers and hyperparameters for similar classification tasks.

The findings emphasize the significance of a structured hyperparameter tuning process in achieving superior model performance and contribute to the ongoing exploration of optimization techniques in machine learning.

Keywords—

I. INTRODUCTION

Machine Learning (ML) models and algorithms are widely used in many application domains including recommendation systems, computer vision, natural language processing, and user behavior analytics.[1] Different ML algorithms are suitable for various types of problems.

Building an effective machine-learning model is a complex and time-consuming process that involves determining the appropriate algorithm and obtaining an optimal model architecture by tuning its hyperparameters.[2]

There are two types of parameters that exist in ML models one is initialized and updated through the data learning process while the other cannot be directly estimated from data.

However, there are still no established methods to optimize hyper-parameters, and various studies are underway.

I.1 Identification of Relevant Contemporary Issue

With the exponential growth of data and the rise of complex machine-learning models, optimizing model performance has become crucial in various applications. [4] Hyperparameter optimization, specifically choosing the right optimizer along with the other hyperparameters, is a contemporary challenge for data scientists and machine learning engineers. Optimizers directly impact model convergence speed, final accuracy, and generalization, making it a critical aspect of machine learning projects.[5]

I.2. Identification of Problem

Selecting the optimal hyperparameters, particularly the optimizer, for machine learning models is a complex task. Improper selection of hyperparameters can lead to slow training, overfitting, or poor generalization. Currently, trial and error and manual tuning are common approaches, which are inefficient and time-consuming.[6] This project focuses on hyperparameter optimization through the comparison of various optimizers, aiming to find the most efficient one for different tasks by implementing Grid search algorithm.

I.3. Identification of Tasks

- Research and review of various optimizers used in machine learning.
- Selection of appropriate datasets for experimentation.
- Analysis and comparison of optimizers based on performance metrics.
- Presentation of findings and recommendations based on results.

Timeline of the Reported Problem

- The concept of hyperparameter optimization has been explored since the early days of machine learning, but its importance has risen significantly with the advent of deep learning. Optimizers like SGD have been used since the 1950s, but more recent adaptive optimizers like Adam (2014) have introduced new efficiencies and challenges. The need for efficient hyperparameter tuning has grown as machine learning models become more complex and resource intensive.[8]
- Several methods for hyperparameter optimization exist, including manual tuning, grid search, and random search. Additionally, modern approaches such as Bayesian optimization and evolutionary algorithms have gained popularity for automating hyperparameter tuning. However, selecting the right optimizer remains a critical task as it directly impacts model performance.[9]

I.4 Existing Solutions

Numerous studies have explored the benefits and limitations of different optimizers:

- **SGD:** Despite its simplicity, it is highly sensitive to the learning rate and tends to converge slowly, especially in complex tasks.[9]
- **Momentum and Nesterov Accelerated Gradient (NAG):** Enhancements to SGD, these optimizers add momentum to accelerate convergence, particularly in scenarios where SGD struggles with local minima.
- **Adam:** Known for its adaptive learning rate and momentum, Adam is often the optimizer of choice for most practitioners. However, recent research has raised concerns about its generalization ability compared to SGD.[6]
- **RMSProp and AdaGrad:** These optimizers also adapt the learning rate, making them more suitable for sparse data problems.

I.5. Problem in Existing Solutions

While optimizers like Adam and RMSProp have gained widespread adoption, their performance varies significantly across different tasks and datasets. Literature lacks comprehensive studies comparing their performance under consistent experimental conditions, which this research aims to address.

II. METHODOLOGY

II.1. Data Collection

We used two datasets in this study:

- **MNIST:** A widely used image classification dataset consisting of 70,000 28x28 grayscale images of handwritten digits [0-9] with corresponding labels.

II.2. Model Architectures

For each dataset, we implemented specific models:

- **MNIST:** A simple Convolutional Neural Network (CNN) was used, comprising two convolutional layers followed by two fully connected layers.
- **Boston Housing:** A Linear Regression model and a Neural Network with three hidden layers were implemented.

II.3. Optimization Technique used

- **Grid Search Optimization Technique:** Grid search (GS) is one of the most widely used methods for exploring hyperparameter configuration spaces [13]. It can be described as an exhaustive exploration technique, or a brute-force method, that evaluates all possible combinations of hyperparameters defined within a grid. GS operates by assessing the Cartesian product of a finite set of values specified by the user [14]. However, GS does not automatically refine its search to exploit well-performing regions of the hyperparameter space. Instead, the following manual process is typically used to identify global optima [13]:

- Begin with a broad search space and coarse-grained intervals for hyperparameter values.
- Based on previous findings of well-performing hyperparameter regions, narrow the search space and refine the intervals.
- Repeat step 2 iteratively until the global optimum is identified.
- Although grid search is simple to implement, its primary limitation lies in its inefficiency in high-dimensional hyperparameter spaces. As the number of hyperparameters increases, the number of evaluations grows exponentially. This exponential growth, often referred to as the **curse of dimensionality** [15], results in computational complexity that scales as

$O(nk)O(n^k)O(nk)$, where k is the number of hyperparameters and n is the number of values each parameter can take [16]. Consequently, GS is an effective hyperparameter optimization (HPO) approach only when the hyperparameter space is limited.

- In practice, tools such as GridSearchCV in the **scikit-learn** library [16] provide a straightforward implementation of the grid search algorithm, enabling efficient detection of optimal hyperparameters. Despite years of research into advanced global optimization methods and numerous alternative HPO techniques, GS remains prevalent for several reasons:

- It is easy to implement and automate.
- It consistently identifies better hyperparameters (λ) compared to manual sequential optimization.
- It is reliable and effective in low-dimensional hyperparameter spaces.

II.4. Optimizers Evaluated

The following optimizers were evaluated:

- **SGD**: SGD with a different learning rate.
- **Adam**: Known for its adaptive learning rate and momentum.
- **RMSProp**: Designed to handle noisy data by adjusting the learning rate based on recent gradients.
- **AdaGrad**: Adapts learning rates individually for each parameter.

III. Experimental Setup

All experiments were conducted using the TensorFlow framework. Each model was trained for 10 epochs, and the following hyperparameters were tuned:

- **Learning rate**: Various learning rates [0.001, 0.01] were tested for each optimizer to ensure fair comparison.
- **Batch size**: Batch sizes of 32, 64 were used across all models.
- **Dense Layers**: Various Numbers of layers, which were 1, 2 were used across all models
- **Neurons per Layer**: The number of perceptron or neurons used in each layer were 64, 128, 256, ensuring that each number of neurons are used for all

possible combinations, which were 96 in total.

III.1. Evaluation Metrics

We used the following metrics to evaluate the performance of the optimizers:

- **Accuracy** for classification tasks (MNIST).
- **Convergence speed**, measured by the number of epochs required to reach a certain accuracy or loss threshold.
- **Generalization**: Tested by evaluating the model on a validation set and recording any overfitting tendencies.

IV. RESULTS AND ANALYSIS

IV.1. Classification Task: MNIST

For the **MNIST dataset**, Adam and RMSprop showed the fastest convergence, but among the two the former came first, reaching almost 98% accuracy with combining it with rest hyperparameters options decided, learning rate 0.001, batch size 32, dense layers 1, Neurons/Layer 256, Training Time 389.25s. **SGD** and **Adagrad** exhibited more fluctuations.

- **RMSProp** performed similarly to Adam in terms of convergence speed but was slightly more stable during training.

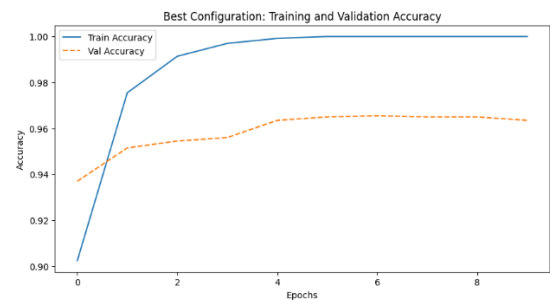


Fig. 4.1.1. Accuracy of Adam

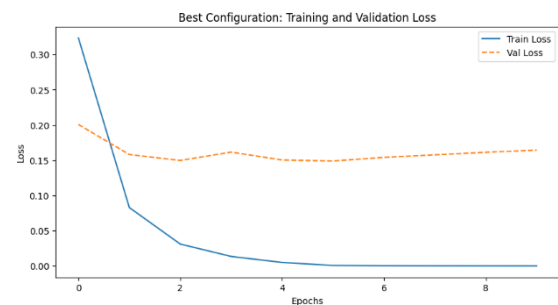


Fig. 4.1.2. Loss by Adam

The total run time to check all the possible combinations, which were 96, was 4 hrs 52 min 29 sec.

This test was performed on cloud platform, mainly Google Colab, and on IDE of the user's PC, on a 64 bit system with 2.9 GHz AMD Ryzen 7 4800H processor, 16GB of memory and 4GB of VRAM.

V. CONCLUSION AND FUTURE WORK

V.1. CONCLUSION

The results indicates that:

- Adam emerged as the best optimizer for the MNIST classification task, offering high accuracy and fast convergence.
- RMSprop provided competitive results.
- SGD and Adagrad were suitable for scenarios prioritizing stability over speed.

Adam's adaptive learning rate helps in achieving better performance in most cases, but simpler optimizers like SGD may offer better generalization. The findings highlight the importance of structured hyperparameters tuning and provide a reference for practitioners in similar tasks.

5.2. FUTURE WORK

Future work will explore:

- The use of hybrid optimizers, complex dataset to strengthen the model
- Implementation of different hyperparameter tuning process such as, Random Forest search optimization, Bayesian optimization, or genetic algorithms.
- Apply the optimizers in larger-scale DL models for more complex tasks.
- Regression dataset, like Boston housing dataset, should be tested to increase the performance of the model.

REFERENCES

- [1] Bielza C, Larrañaga P. Title: Estimation of Distribution Algorithms for Generative Adversarial and Con-volutional Neural Network Hyper-Parameter Optimization. 2024.
- [2] Kingma DP, Lei Ba J. ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION.
- [3] Bischl B, Binder M, Lang M, Pielok T, Richter J, Coors S, et al. Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. Vol. 13, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. John Wiley and Sons Inc; 2023.
- [4] Negrin I, Roose D, Chagoyén E, Leuven KU. PARAMETER TUNING STRATEGIES FOR METAHEURISTIC METHODS APPLIED TO DISCRETE OPTIMIZATION OF STRUCTURAL DESIGN.
- [5] Wilson AC, Roelofs R, Stern M, Srebro N, Recht B. The Marginal Value of Adaptive Gradient Methods in Machine Learning.
- [6] Bergstra J, Bardenet R, Bengio Y, Kégl B. Algorithms for Hyper-Parameter Optimization.
- [7] Por R, Luque Á, Tutorizado L, Rubio EL, García González J. Hyperparameter optimization for the radiance field reconstruction model DirectVoxGo.
- [8] Hazan E, Klivans A, Yuan Y. Hyperparameter Optimization: A Spectral Approach. 2017 Jun 2.
- [9] Yang L, Shami A. On hyperparameter optimization of machine learning algorithms: Theory and practice. Neurocomputing. 2020 Nov 20.
- [10] Andonie R. Hyperparameter optimization in learning systems. Journal of Membrane Computing. 2019 Dec 1.
- [11] Zoller M-A, Huber MF. Benchmark and survey of automated machine learning frameworks, arXiv preprint arXiv: 1904.12054, 2019.
- [12] Hutter F, Kotthoff L, Vanschoren J. Automated machine learning: methods, systems, challenges. Springer Nat. 2019;1:219.
- [13] Claesen M, Simm J, Popovic D, et al. Easy hyperparameter search using optunity, arXiv preprint arXiv: 1412.1114, 2014.
- [14] Daniel Mesafint Belete & Manjaiah D. Huchaiah (2021): Grid search in hyperparameter optimization of machine learning models for prediction of HIV/AIDS test results.