TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ MÔN INTRODUCTION TO MACHINE LEARNING**

**FINAL REPORT**

*Người hướng dẫn*: **GV LÊ ANH CƯỜNG**

*Người thực hiện*: **PHẠM PHÚ BÌNH – 521H0495**

Lớp **: 21H50302**

Khoá  **: 25**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN GIỮA KÌ MÔN INTRODUCTION TO MACHINE LEARNING**

**FINAL REPORT**

*Người hướng dẫn*: **GV LÊ ANH CƯỜNG**

*Người thực hiện*: **PHẠM PHÚ BÌNH**

Lớp **: 21H50301**

Khoá  **: 25**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

To complete this mid-term project, we would like to express our gratitude to Ton Duc Thang University for providing the necessary facilities and to the teachers who have supported us throughout our studies at the university. Moreover, we love to give thanks to our parents who gave us chances to approach knowledge at TDTU and motivated, supported us a lot.

Especially, we would like to thank Mr. Le Anh Cuong for teaching us with great dedication and detail, so that we have enough knowledge to use for this essay. Due to our limited experience and knowledge, we are sure that there are some mistakes in our work. we sincerely hope to receive feedbacks and constructive criticism from our teacher so that we can complete this essay more effectively.

We would like to express my heartfelt thanks and wish teacher good health.

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Chúng tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của TS Lê Anh Cường;. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 20 tháng 12 năm 2023*

*Tác giả*

Phạm Phú Bình

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

This research provides a detailed analysis of optimizer methods in machine learning model training, specifically comparing algorithms such as Stochastic Gradient Descent (SGD), Adam, and RMSprop. The aim is to guide practitioners in selecting the most suitable optimizer for enhanced model performance based on their dataset characteristics.

Additionally, the study explores the integration of Continual Learning and Test Production in machine learning solutions. Continual Learning ensures model adaptability to evolving data, while Test Production focuses on robust testing mechanisms for evaluating model performance and generalization.

By combining theoretical exploration with practical insights, the research addresses the dynamic nature of machine learning systems. It offers valuable perspectives on the challenges and opportunities associated with incorporating Continual Learning and Test Production into a holistic solution. The ultimate goal is to equip researchers and practitioners with the knowledge needed to optimize model training processes, improve adaptability to changing data scenarios, and contribute to the development of more effective machine learning solutions.

MỤC LỤC

[LỜI CẢM ƠN 1](#_Toc154184752)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN 3](#_Toc154184753)

[TÓM TẮT 4](#_Toc154184754)

[CHAPTER 1 – THE GOAL OF CREATING MACHINE LEARNING MODEL 6](#_Toc154184755)

[1. What is the Optimizer? 6](#_Toc154184756)

[1.1 Definition of optimizers 6](#_Toc154184757)

[1.2 The purpose of using optimizers 6](#_Toc154184758)

[2. Types of Optimizers and Comparison 7](#_Toc154184759)

[2.1 Gradient Descent 7](#_Toc154184760)

[2.2 Stochastic Gradient Descent (SGD) 9](#_Toc154184761)

[2.3 Momentum 11](#_Toc154184762)

[2.4 Aragrad 12](#_Toc154184763)

[2.5 RMSprop 13](#_Toc154184764)

[2.6 Adam 14](#_Toc154184765)

[CHAPTER 2 - CONTINUAL LEARNING AND TEST PRODUCTION 15](#_Toc154184766)

[1 CONTINUAL LEARNING 15](#_Toc154184767)

[1.1 Definition 15](#_Toc154184768)

[1.2 Continual learning is often misinterpreted: 16](#_Toc154184769)

[1.3 The Four Stages of Continual Learning: 16](#_Toc154184770)

[1.4 Challenger: 17](#_Toc154184771)

[2. Testing Production 18](#_Toc154184772)

[2.1 Definition 18](#_Toc154184773)

[2.2 Purpose of Testing Production: 18](#_Toc154184774)

[2.3 Testing Production Strategies: 18](#_Toc154184775)

[TÀI LIỆU THAM KHẢO 21](#_Toc154184776)

CHAPTER 1 – THE GOAL OF CREATING MACHINE LEARNING MODEL

1. What is the Optimizer?

1.1 Definition of optimizers

Optimizers is a algorithms that use in the machine learning and deep learning ( the subfield of machine learning)

In machine learning or deep learning, we always need an algorithm finds the value of the parameters (weights) that minimize the error when mapping inputs to outputs and an optimizer is a algorithm that perform minimize the error.

Optimizers is the foundation for neural networks to learn from data by iteratively updating weights and biases. Common optimizers include Stochastic Gradient Descent (SGD), Adam, and RMSprop. Each optimizer has specific update rules, learning rates, and momentum to find optimal model parameters for improved performance.

**1.2** The purpose of using optimizers

Optimizer algorithms are optimization method that helps improve a deep learning model’s performance. Optimizers play a crucial role in minimizing the loss function or error between the predicted output and the actual target values during the training phase. Some key purpose of using optimizers:

* **Faster Convergence:** Enabling models to achieve optimal performance with fewer iterations. Efficient optimization methods accelerate learning from data.
* **Avoiding Local Minima:** Navigate parameter space to evade local minima, facilitating the discovery of the global minimum for a more accurate and robust model.
* **Adaptive Learning Rates:** Modern optimizers like Adam adapt learning rates during training, leading to faster convergence in steep gradient directions and slower convergence in flatter regions, enhancing overall optimization performance.
* **Handling Noisy or Sparse Data:** Adjust learning rates for individual parameters to handle noisy or sparse datasets, preventing overshooting or slow convergence in the presence of irregular data patterns.
* **Parallelization and Scalability:**

Optimizers are vital for parallelizing the training process, enabling the training of complex models on large datasets through distributed computing, crucial for handling big data in machine learning applications.

2. Types of Optimizers and Comparison

2.1 Gradient Descent

Gradient Descent is an iterative optimization algorithm used in Machine Learning and Deep Learning, particularly in convex optimization problems, with the goal of finding a set of internal parameters for optimizing models

Gradient Descent comes in various forms, such as Stochastic Gradient Descent (SGD), Mini-batch SGD. However, fundamentally, they are executed as follows:

Initialize internal variables.

Evaluate the model based on internal variables and the loss function.

Update internal variables in the direction that optimizes the loss function (finding optimal points).

Repeat steps 2 and 3 until the stopping condition is met.

The update formula for Gradient Descent can be written as:

A screenshot of a computer

Description automatically generated

An example of GD:

Imagine you are lost on a mountain in dense fog, and you can only sense the slope of the ground beneath your feet. The most efficient way to descend quickly is to follow the steepest slope. This is precisely the concept implemented by Gradient Descent. At each point on the function, it determines the slope and then moves in the opposite direction of the slope until the slope becomes zero (a minimum).

In essence, it mimics the idea of finding the quickest way downhill in a foggy environment. The algorithm iteratively adjusts the variables in the direction of the steepest slope, updating them until it reaches a point where the slope is zero, indicating a minimum. This downhill descent is analogous to navigating the foggy mountain slope by always moving towards the steepest downward direction until reaching a valley or a flat area.

In real-situation, the slope just be nearest zero

A diagram of a weight loss

Description automatically generated

Image of Gradient Descent

* **Advantages:**
* Gradient Descent is a fundamental and easy-to-understand algorithm. It effectively addresses the optimization of neural network models by updating weights after each iteration.
* **Disadvantages:**
* Due to its simplicity, Gradient Descent has limitations, such as dependency on the initial solution and learning rate.
* For example, in a function with two global minima, the algorithm's outcome depends on the choice of initial points, resulting in different final solutions.
* Excessive learning rates can prevent the algorithm from converging, causing it to oscillate around the target, while very small learning rates impact training speed.

2.2 Stochastic Gradient Descent (SGD)

Stochastic Gradient Descent (SGD) is a variant of Gradient Descent. Instead of updating weights once after each epoch, in each epoch with N data points, we update weights N times. On one hand, SGD reduces the speed of a single epoch. However, from a different perspective, SGD converges very quickly, typically within a few epochs. The formula for SGD is similar to GD but is applied to each data point individually.

A diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of

Description automatically generated

SGD follows a zigzag path,GD follows a straight. To comparison, SGD not as smooth as GD. This is understandable because a single data point cannot represent the entire dataset.

GD has limitations for large databases (several million data points) due to the cumbersome computation of derivatives over the entire dataset in each iteration. Additionally, GD is not suitable for online learning. Online learning is when data is continuously updated (e.g., adding new registered users), and every time new data is added, we have to recalculate the derivatives over the entire dataset, which leads to slow computation, making the algorithm no longer online.

Therefore, SGD was introduced to address this issue, as it only needs to update on the new data point, suitable for online learning.

An illustrative example: with 10,000 data points, SGD can achieve a good solution after just 3 epochs, whereas GD would require up to 90 epochs to achieve the same result.

* **Advantages:**
* This algorithm effectively handles large databases, a task GD struggles with. SGD is still widely used today.
* **Disadvantages:**
* The algorithm has not yet resolved the two major shortcomings of gradient descent (learning rate, initial data point). Thus, it needs to be combined with other algorithms like Momentum, AdaGrad, etc. These algorithms will be discussed in the following section.

2.3 Momentum

Gradient Descent with Momentum is a variant of the Gradient Descent algorithm designed to overcome certain limitations of GD, such as slow convergence and susceptibility to getting stuck in local minima.

In this algorithm, instead of solely relying on the current gradient to update weights, a momentum term is introduced, representing a fraction of the previous gradient, into the updating process. Specifically, the weight update formula in Gradient Descent with Momentum is modified as follows:

A screenshot of a computer

Description automatically generated

* **Advantages:**
* Momentum helps accelerate the convergence of the optimization process by reducing oscillations and avoiding getting stuck in local minima.
* **Disadvantages:**
* In some cases, Momentum might introduce oscillations, particularly when approaching the minimum. This behavior can slow down convergence or cause instability.
* The performance of Momentum is sensitive to the choice of the momentum coefficient (typically denoted as γ). Improper tuning of this hyperparameter can lead to suboptimal results.

2.4 Aragrad

In contrast to preceding algorithms where the learning rate is typically constant throughout training, Adagrad treats the learning rate as a dynamic parameter, allowing it to vary at each time step.

A black background with white text

Description automatically generated

* **Advantages:**
* A clear advantage of Adagrad is its ability to automatically adjust the learning rate, eliminating the need for manual tuning.
* **Disadvantages:**
* One drawback is that Adagrad's sum of squared gradients tends to increase over time, resulting in an extremely small learning rate that can effectively halt the training process.
* In summary, Adagrad's automatic learning rate adaptation is a positive feature, but its tendency to excessively reduce the learning rate over time may hinder training.

2.5 RMSprop

RMSprop solves the issue of the diminishing learning rate in Adagrad by dividing the learning rate by the average of the squared gradients.

A mathematical equation with numbers and symbols

Description automatically generated

* **Advantages:**
* The most notable advantage of RMSprop is its resolution of the diminishing learning rate problem observed in Adagrad. The diminishing learning rate issue over time can lead to a progressively slower training, potentially resulting in freezing.
* **Disadvantages:**
* RMSprop may converge to a local minimum rather than reaching the global minimum, unlike Momentum. Therefore, practitioners often combine both algorithms, Momentum and RMSprop, to create an optimized algorithm known as Adam.

**2.6 Adam**

Adam's adaptive learning rates and combination of Momentum and RMSprop make it a powerful optimizer. Adam requires careful hyperparameter tuning and may not always be the optimal choice for every optimization problem.

Image using adam algorithm as a heavy and accerately ball, it solve the problem that approach to the global minimum, not local minimum and also effective solutions to issues like slow convergence.

Here is the formula of Adam

A math equations on a white background

Description automatically generated

* **Advantages:**
* Solve both the problems of Momentum and RMSprop algorithm
* Performs well with sparse gradients and is robust in scenarios with noisy or sparse data.
* Disadvantages:
* Sensitive to the choice of hyperparameters, including the learning rate, beta1, and beta2. Improper tuning may result in suboptimal convergence.
* Not always be the optimal choice for every scenario. In some cases, simpler optimizers like SGD or variants may perform better.

CHAPTER 2 - CONTINUAL LEARNING AND TEST PRODUCTION

1 CONTINUAL LEARNING

1.1 Definition

Continual Learning is the idea of updating your model as new data becomes available; this makes your model keep up with the current data distributions.

Once your model is updated, it cannot be blindly released to production. It needs to be tested to ensure that it is safe and that it is better than the current model in production

1.2 Continual learning is often misinterpreted:

Continual learning is NOT referring to a special class of ML algorithms that allow for incremental update of the model when every single new datapoint becomes available. It is not a “online learning algorithm”

Continual learning does NOT mean starting a retraining job every time a new data sample becomes available.

**Extra: Purpose of Continual Learning:**

The base reason for it is to help your model keep up with data distribution shifts.

1.3 The Four Stages of Continual Learning:

* **Stage 1: Manual, stateless retraining**

In this initial stage, retraining is a manual process, and the system lacks the capability to adapt autonomously. When new data becomes available, human intervention is required to initiate the retraining process. The model is stateless, meaning it doesn't retain information about its previous training sessions.

* **Stage 2: Fixed schedule automated stateless retraining**

Progressing from manual retraining, the system moves to an automated process with a fixed schedule. Retraining is scheduled at regular intervals, often based on predefined timeframes (e.g., daily, weekly). While automation reduces human intervention, the system remains stateless, lacking memory of past training sessions.

* **Stage 3: Fixed schedule automated stateful training**

At this stage, the system advances to stateful training, meaning it retains information about previous training sessions. The model incorporates memory of its past states and adapts to changes in the data distribution. Automated retraining still occurs at fixed intervals, but the system has a form of historical context.

* **Stage 4: Continual learning**

The most advanced stage, continual learning involves dynamic adaptation to new information in real-time. Models in this stage can learn incrementally from each new data point, adapting and updating continuously without fixed schedules. These models are capable of handling concept drift, evolving over time, and retaining knowledge gained from previous experiences. Continual learning systems can autonomously trigger retraining based on performance metrics or changes in the data distribution.

1.4 Challenger:

**Fresh Data Access Challenge:**

1. Varied data sources depositing data at different speeds.
2. Dependency on external vendor systems for fresh data.
3. Transitioning from batched ETLs to real-time transport processing.

**Speed of Labeling:**

1. Utilizing natural labels with short feedback loops.
2. Exploring weak-supervision or semi-supervision techniques.
3. Addressing label computation speed, considering both batch and streaming approaches.

**Evaluation Challenge:**

Increased frequency of model updates heightens the risk of catastrophic failures.

**Data Scaling Challenge:**

1. Stateful training requiring tracking of global statistics.
2. Incremental calculation of statistics to adapt to new data.

**Algorithm Challenge:**

1. Inability to incrementally train these models at a fast update frequency.
2. Example: PCA dimensionality reduction requiring the full dataset.
3. Limited adoption of incrementally trainable variants like Hoeffding Trees.

2. Testing Production

2.1 Definition

Testing in production (TiP) is a methodology in machine learning (ML) and software development that involves performing testing activities directly in a live, production environment rather than in isolated testing environments. In the context of machine learning, TiP allows practitioners to evaluate and validate models using real-world data and user interactions.

2.2 Purpose of Testing Production:

* Real-world validation of model performance.
* Immediate feedback on user interactions and business impact.
* Detection of issues that might only surface in a live environment.
* Continuous improvement and adaptation to changing conditions.

2.3 Testing Production Strategies:

**Shadow Deployment:**

**Pros:**

Safest deployment; buggy models won't affect predictions.

Conceptually simple with fast statistical significance.

**Cons:**

Incompatible with observing user interactions.

Doubles prediction count, increasing computational cost.

**A/B Testing:**

**Pros:**

Allows capturing user reactions to different models.

Simple and cost-effective with one prediction per request.

**Cons:**

Less safe; requires strong offline evaluation.

Balancing risk vs. sample size for analysis.

**Canary Release:**

**Pros:**

Easy to understand and implement with existing feature flagging.

Cost-effective with incremental traffic shift.

**Cons:**

Prone to less rigorous performance determination.

Potential accidents without careful supervision.

Interleaving Experiments:

**Pros:**

Identifies the best model with smaller sample size.

Captures user behavior with full traffic.

**Cons:**

Complex implementation compared to A/B testing.

Doubles compute power and limited scalability.

**Bandits:**

**Pros:**

Requires less data for model performance determination.

More data-efficient and minimizes opportunity cost.

**Cons:**

Complex implementation with continuous feedback.

Applicable only to certain use cases; not as safe as Shadow Deployments.

TÀI LIỆU THAM KHẢO

[1] serodriguez68. Course “Chapter 9 - Continual learning and test in production”. Dec 12, 2021. <https://github.com/serodriguez68/designing-ml-systems-summary/blob/main/09-continual-learning-and-test-in-production.md#continual-learning-challenges> . Address Dec 21 ,2023.

[2] Niklas Donges. Course “Gradient Descent in Machine Learning: A Basic Introduction." <https://builtin.com/data-science/gradient-descent> Address Dec 20,2023.

[3] Trần Trung Trực. Course “Optimizer- Hiểu sâu về các thuật toán tối ưu ( GD,SGD,Adam,..)”. September 29, 2020 . <https://viblo.asia/p/optimizer-hieu-sau-ve-cac-thuat-toan-toi-uu-gdsgdadam-Qbq5QQ9E5D8> .Address Oct 20,2023.

[4] Nguyễn Đình Quý. Course “Gradient Descent”.2021. <https://ndquy.github.io/posts/gradient-descent-2/> .Address Dec 20,2023.

[5] GeekforGeeks Course “Gradient Descent Algorithm in Machine Learning” <https://www.geeksforgeeks.org/gradient-descent-algorithm-and-its-variants/> .Address Dec 20,2023.