Vinícius Yuiti Fukase

Critical Learning Periods: Identifying Moments That Matter in Deep Neural Networks

São Paulo, SP

2026

Vinícius Yuiti Fukase

Critical Learning Periods: Identifying Moments That Matter in Deep Neural Networks

Final Version

Dissertation submitted to the Graduate Program in Electrical Engineering at Escola Politécnica da Universidade de São Paulo in partial fulfillment of the requirements for the degree of Master of Science.

Advisor: Prof. Dr. Artur Jordão Lima Correia

São Paulo, SP

2026

Catalogação-na-publicação

Fukase, Vinícius
Critical Learning Periods: Identifying Moments That Matter in Deep Neural Networks / V. Fukase -- São Paulo, 2026.
64 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.INTELIGÊNCIA ARTIFICIAL 2.APRENDIZADO COMPUTACIONAL 3.APRENDIZAGEM PROFUNDA 4.REDES NEURAIS I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

Elaborada pela Divisão de Biblioteca da Escola Politécnica da USP, por meio de formulário eletrônico, com os dados fornecidos pelo(a) autor(a)

Nome: Vinícius Yuiti Fukase

Título: Períodos Críticos de Aprendizado: Identificando Momentos que Importam em Redes Neurais Profundas

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Ciências - Programa de Pós-Graduação em Engenharia Elétrica

Aprovado em: 30 de junho de 2026

Banca Examinadora:

Prof. Dr.: Artur Jordão Lima Correia

Instituição: Universidade de São Paulo

Julgamento: Aprovado _____

Prof. Dr(a).: Aline Marins Paes Carvalho

Instituição: Universidade Federal Fluminense

Julgamento: Aprovado _____

Prof. Dr(a).: Leo Sampaio Ferraz Ribeiro

Instituição: Universidade de São Paulo

Julgamento: Aprovado _____

Acknowledgements

I would like to express my profound gratitude to my family, Marcena and Noboru, for their unwavering support throughout this journey. My wife Gabriela deserves special thanks for her constant presence and assistance whenever it was most needed.

I am deeply indebted to Professor Artur Jordão Lima Correia for his exceptional guidance and mentorship during my master's studies.

My heartfelt thanks go to my peers at Universidade de São Paulo, with a special mention of Barbara Fernandes Dias Bueno, Heitor Gama Ribeiro and Lucas Martins Libanio for their invaluable contributions to this thesis. Additionally, I extend my sincere appreciation to all members of the research group for their collaborative spirit and support.

Resumo

FUKASE, V. Y. **Períodos Críticos de Aprendizado: Identificando Momentos que Importam em Redes Neurais Profundas.** 2026. Dissertação (Mestrado) — Escola Politécnica da Universidade de São Paulo, São Paulo, 2026.

Períodos críticos de aprendizagem compreendem um fenômeno importante que envolve a aprendizagem profunda, uma fase em que a regularização e a densidade dos dados determinam a capacidade final de generalização do modelo. Trabalhos existentes nesta linha de pesquisa confirmam a existência desse fenômeno e fornecem percepções úteis. No entanto, a literatura carece de esforços para identificar com precisão quando ocorrem os períodos críticos. Neste trabalho, preenchemos essa lacuna com a introdução de uma abordagem sistemática para identificar períodos críticos durante o treinamento de redes neurais profundas, concentrando-nos na eliminação de técnicas de regularização computacionalmente intensivas e na aplicação eficaz de mecanismos para reduzir os custos computacionais, como a poda de dados. A ideia central do trabalho aproveita mecanismos de previsão de generalização para identificar as fases críticas em que as receitas bem definidas de treinamento produzem benefícios máximos para o desempenho da capacidade preditiva dos modelos. Ao interromper as receitas que demandam muitos recursos computacionais além desses períodos, aceleramos significativamente a fase de aprendizado e conseguimos reduções significativas em métricas associadas à performance computacional como, por exemplo, tempo de treinamento, consumo de energia e emissões de CO_2. Ao contrário dos métodos de poda de dados de última geração, que apresentam instabilidade–muitas vezes reintroduzindo amostras podadas para recuperar a generalização do modelo–nosso método garante uma redução estável e monotônica ao respeitar o período crítico. Validamos nossa abordagem em uma ampla gama de arquiteturas em *benchmarks* padrão. Resultados empíricos demonstram que nossa abordagem alcança um desempenho preditivo quase sem perdas, limitando a queda média na precisão a apenas **0,09%** (contra **0,26%** para métodos de última geração), enquanto mantém eficiência de treinamento equivalente. Nosso trabalho aumenta a compreensão da dinâmica de treinamento e abre caminho para práticas de aprendizagem profunda mais sustentáveis e eficientes, especialmente em ambientes com recursos limitados. Na era da corrida por grandes modelos, acreditamos que nosso método surge como um mecanismo valioso.

Palavras-chave: Aprendizagem Profunda, Redes Neurais, Períodos Críticos, Regularização, IA Verde.

Abstract

FUKASE, V. Y. **Critical Learning Periods: Identifying Moments That Matter in Deep Neural Networks.** 2026. Dissertation (Master's) — Escola Politécnica da Universidade de São Paulo, São Paulo, 2026.

Critical learning periods comprehend an important phenomenon involving deep learning, a phase where regularization and data density determine the final generalization capacity of the model. Existing works confirm the existence of this phenomenon and provide useful insights. However, the literature lacks efforts to precisely identify when critical periods occur. In this work, we fill this gap by introducing a systematic approach for identifying critical periods during the training of deep neural networks, focusing on eliminating computationally intensive regularization techniques and effectively applying mechanisms for reducing computational costs such as data pruning. Our method leverages generalization prediction mechanisms to pinpoint critical phases where training recipes yield maximum benefits to the predictive ability performance of models. By halting resource-intensive recipes beyond these periods, we significantly accelerate the learning phase and achieve significant reductions in training time, energy consumption, and CO_2 emissions. Unlike state-of-the-art data pruning methods that exhibit instability–often re-introducing pruned samples to recover model generalization–our method ensures a stable, monotonic reduction by respecting the critical period. We validate our approach in a broad range of architectures on standard benchmarks. Empirical results demonstrate our approach achieves near-lossless model generalization performance, limiting the mean accuracy drop to just **0.09%** (vs. **0.26%** for top-performance methods), while maintaining equivalent training efficiency. Our work enhances understanding of training dynamics and paves the way for more sustainable and efficient deep learning practices, particularly in resource-constrained environments. In the era of the race for foundation models, we believe our method emerges as a valuable framework.

Keywords: Deep Learning, Neural Networks, Critical Periods, Regularization, Green AI.

List of Figures

List of Tables

List of abbreviations and acronyms

AI	Artificial Intelligence
BWS	Best Window Selection
CNN	Convolutional Neural Network
CO_2	Carbon dioxide
IES	Instance-dependent early stopping
LLM	Large Language Models
LoRA	Low-Rank Adaptation
RSRS	Repeated Sampling of Random Subsets
SGD	Stochastic Gradient Descent

Contents

1 Introduction

Deep learning continues to advance in various cognitive tasks and serves as the primary engine for learning patterns from data (BENGIO et al., 2025; SAJADIEH et al., 2026). Within this sphere, architectures such as Convolutional Neural Networks (CNNs) and Vision Transformers represent the core pillars of research in image classification (MAURÍCIO et al., 2023). Beyond these specific models, recent trends favor increasingly large systems, often known as foundation models (BOMMASANI et al., 2021). These systems act as a primary starting point for learning novel downstream applications via large-scale optimization.

While large models yield impressive results, their high computational costs trigger substantial environmental impacts (FAIZ et al., 2024). For example, training a high-parameter model family requires up to 16,000 GPUs in a global distribution (GRATTAFIORI et al., 2024). Recent estimates place the training costs of top-tier models at approximately 79 million dollars and 129 million dollars, respectively (SAJADIEH et al., 2026). Beyond the financial difficulty, the carbon footprint remains extreme; the training process generates thousands of tons of CO_2 (SAJADIEH et al., 2026). For example, the carbon emission from training even an 8-billion parameter model equals 83 years of energy consumption for a typical US home, demonstrating the importance of holistically evaluating the environmental impact of creating language models (MORRISON et al., 2025).

The previous trend toward massive scaling introduces significant barriers for researchers with finite infrastructure. Despite gains in efficiency, state-of-the-art models continue to grow in parameter count, increasing both latency and power demands. These factors shift the focus of the scientific community toward sustainable AI development, a field known as GreenAI (SCHWARTZ et al., 2020).

To address the previous challenges, researchers focus on reducing training costs (LACOSTE et al., 2019; MORRISON et al., 2025). In this direction, strategies such as dataset distillation (YU et al., 2023; CAZENAVETTE et al., 2022), offer potential solutions by reducing training size through synthetic samples; however they frequently incur significant computational overheads or fail to match the accuracy of the full dataset (PONS et al., 2025). In contrast, data pruning targets a more efficient balance (QIN et al., 2024; YUAN et al., 2025), aiming to achieve lossless generalization with minimal extra cost. By identifying and discarding samples that provide negligible learning information, data pruning lowers the computational budget while maintaining the final model quality.

Despite promoting positive results, existing state-of-the-art data pruning methods typically apply pruning heuristics statically, often ignoring the critical early phases of

learning dynamics (QIN et al., 2024; YUAN et al., 2025). In particular, these strategies frequently fail to account for the dynamic nature of learning, often removing samples that appear redundant or unimportant simply because the model did not learn them. Consequently, the model lacks vital information required to shape its early optimization path (QIN et al., 2024; YUAN et al., 2025). Figure 1 corroborates this behavior on ResNet18 architecture on CIFAR-10. The figure reveals an instability in the Instance-dependent Early Stopping (IES) (YUAN et al., 2025) method (blue line): in later epochs, it re-introduces previously pruned samples—increasing the training set size again—in an attempt to recover lost predictive performance. Note that this behavior occurs at different epochs (e.g., 91, 160-200).

In a parallel line of research, recent studies exploit the critical learning period (ACHILLE et al., 2019; KLEINMAN et al., 2024). These periods refer to a deep learning phenomenon where early epochs determine success in many training recipes, including regularization and the capacity of the model to combine information from diverse sources (ACHILLE et al., 2019; KLEINMAN et al., 2023). Understanding critical learning periods in deep learning significantly boosts training efficiency and overall model performance. Despite their acknowledged significance, accurately identifying such periods during the training phase continues to be elusive. These periods profoundly influence learning dynamics and effectiveness. Therefore, systematically identifying them becomes essential to optimize training efficiency and model generalization (BALLESTER et al., 2024).

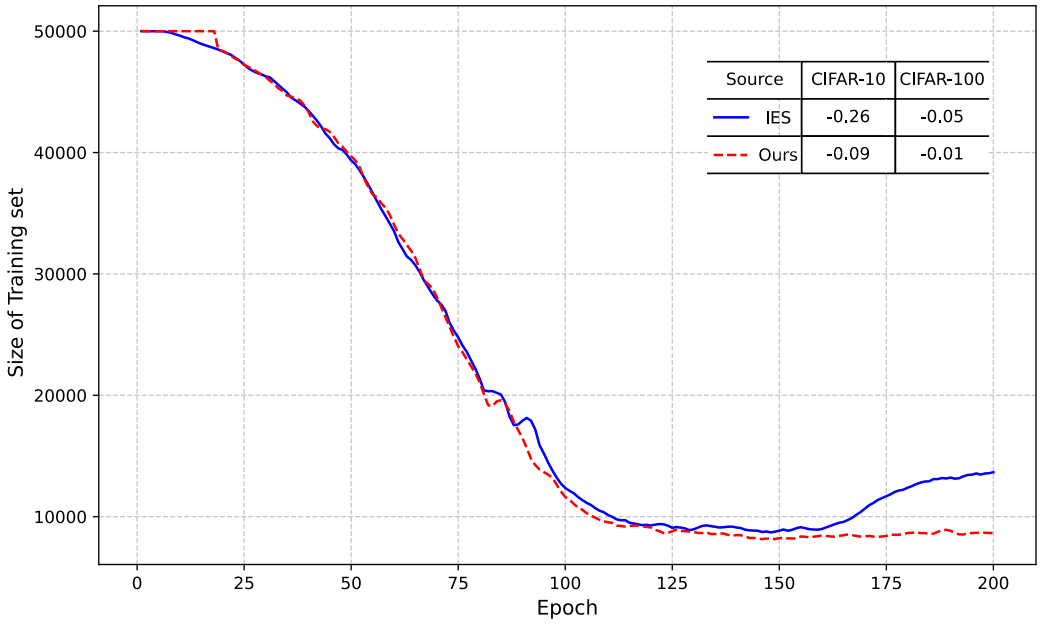


Figure 1 – Impact of data pruning timing on training dynamics. The curves show the training set size evolution for a modern state-of-the-art data pruning method, IES (YUAN et al., 2025) (blue) vs. our method (red) on ResNet18 architecture.

Prior research confirms that critical periods manifest early in the training process, beyond which numerous training methods yield minimal to no additional advantage (ACHILLE et al., 2019). Kleinman et al. 2023 emphasize that the capacity of the neural network to combine data from varied origins significantly depends on their exposure to appropriately correlated stimuli during initial training stages. Intricate and unpredictable early transient dynamics illustrate the emergence of critical periods. These dynamics play a crucial role in determining final efficacy and the nature of representations a system acquires upon training completion. The seminal work by Kleinman et al. 2024 demonstrates that learning ability does not increase monotonically during training and that a memorization phase occurs in early epochs, where models retain most discriminative information. Furthermore, the authors show that critical periods occur during the memorization phase and, after this, the model may start to weaken its ability to retain knowledge (forgetting phase).

1.1 Motivation

Existing studies argue that the effectiveness of regularization during model training extends beyond merely preventing solution entrapment in local minima (ACHILLE et al., 2019). Specifically, they find that adjusting regularization practices after a critical period in training changes weight values and thereby the position of the model in the loss landscape. Consistent generalization confirms that this adjustment does not improve the predictive ability of models. Rather, its importance lies in guiding initial stages of training towards areas of loss landscape that are rich in diversity, yet equally effective solutions with strong generalization characteristics. Although these works pose an important role in critical learning periods, none of them offer a systematic way to identify critical periods. It is worth mentioning that Golatkar et al. 2019 find that critical periods emerge early in training but not exactly when (i.e., the epoch). Therefore, a natural question that arises is:

How to identify critical periods during the course of training?

Given calls for more efficient training in the era of foundation models (BENGIO et al., 2025), answering this question yields two notable gains. Firstly, one applies strong regularization only while the model is apt to absorb information (during the critical periods). Secondly, one reduces numbers of training examples through data pruning only after the critical period; thus, speeding-up the learning phase while preventing loss in predictive ability. Figure 2 illustrates this scenario, where we apply strong regularization during critical periods, and then we remove it. These strategies enable better allocation of computational resources, avoiding unnecessary demands, and significantly speeding up the training process. In addition, they also directly contribute to the urgent dialogue surrounding Green AI.

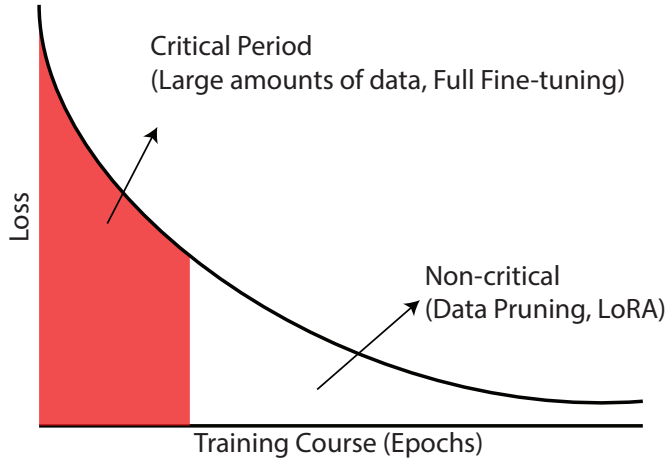


Figure 2 – Evolution of the loss function over epochs, showcasing the application of strong regularization during key periods, followed by a decrease in the number of training samples. This conceptual illustration underscores the strategic shift from 'extensive data utilization/full fine-tuning' to 'data pruning/Low-Rank Adaptation (LoRA)', thereby preserving learning efficacy while minimizing the consumption of computational resources.

Furthermore, the data pruning domain presents a secondary question: *When is the suitable moment to start data pruning without compromising learning effectiveness?* Determining this effective onset is crucial; removing data too early risks discarding data before the model captures essential patterns, while pruning too late diminishes the computational gains. Therefore, identifying the boundary between the need of full-data training and the safety of data reduction becomes the central challenge for existing data pruning methods.

1.2 Objectives

The objectives of this research focus on optimizing the training phase of deep image classification models by leveraging critical learning periods. Our objective is threefold, encompassing both theoretical understanding and practical application:

Theoretical Objectives

- To underscore the significance of pinpointing the exact moment or epoch when critical periods in training emerge.
- To introduce a systematic method for identifying these crucial moments within the training process.
- To investigate the underlying dynamics of generalization behavior in deep neural networks related to critical periods.

Practical Objectives

- To demonstrate the possibility of substantially reducing computational expenses by strategically adjusting training recipes after critical learning periods across diverse image classification architectures.
- To promote more efficient ways to train large deep learning models.
- To promote greater accessibility for the scientific community, particularly those lacking extensive infrastructure and hardware, to develop and keep pace with innovations in deep learning.

1.3 Contributions

This dissertation organizes the primary contributions into two distinct areas, with the first exploring critical learning periods:

- A systematic method to identify critical learning periods in deep neural network training.
- Achieve significant reductions in training time, energy consumption, and CO_2 emissions by leveraging these strategies. Specifically, experiments demonstrate reductions in training time of popular architectures by up to 59.67%, leading to a 59.47% decrease in CO_2 emissions and a 60% reduction in financial costs.
- Enhance the understanding of training dynamics and pave the way for more sustainable and efficient deep learning practices, especially in resource-constrained environments.

Following these findings, our second contributions are:

1. First investigation on the effective onset for data pruning, shifting the focus from what to prune to when to prune, moving beyond the limitation of initiating data pruning before the model captures essential data patterns.
2. A strategy that explicitly aligns data reduction with critical learning periods, ensuring effective information learning before starting to remove data.
3. Outperform state-of-the-art methods on standard benchmarks, achieving comparable training time speed but superior stability and higher accuracy. Furthermore, by simply coupling our orthogonal approach to existing techniques (i.e. dynamic random pruning) pushing these boundaries even further, demonstrating that preserving critical learning periods is a fundamental key to data efficiency.

4. Achieve significant reductions in training time, energy consumption, and CO_2 emissions by leveraging these strategies. Specifically, experiments demonstrate reductions in training time of popular architectures by up to 42%.

These contributions culminate in the publication of the following paper during this research:

1. *One Period to Rule Them All: Identifying Critical Learning Periods in Deep Networks* (FUKASE et al., 2025a), published in the Procedia Computer Science 2025.
2. *Towards Efficient Training through Critical Periods* (FUKASE et al., 2025b), published in the Workshop of Works in Progress (WiP) - Conference on Graphics, Patterns and Images (SIBGRAPI) 2025.
3. *When to Prune? The Importance of Timing in Data Efficiency Training* (FUKASE et al., 2026), published in the International Conference on Pattern Recognition (ICPR) 2026.

To promote reproducibility from our experiments to the scientific community, we release the source code at github.com/ViniFukase/WhenToPrune.

1.4 Research Statement

Throughout this research, we confirm the following statement:

Throughout the course of training, a simple generalization estimation enables successfully identifying when the critical period emerges. Once we establish this period, we determine the suitable moment to initiate data pruning. By restricting data pruning to this safe zone, the model retains essential information from the start, removing the necessity to re-assimilate unlearned data that is inherent to heuristic approaches.

We confirm the previous statement across a broad range of benchmarks EuroSat (HELBER et al., 2018), (CIFAR-10/100 (KRIZHEVSKY et al., 2009a; KRIZHEVSKY et al., 2009b), Tiny ImageNet (LE; YANG, 2015) and ImageNet30 (HENDRYCKS et al., 2019)).

1.5 Work Organization

We organize the remainder of this dissertation as follows. Chapter 2 provides the necessary background on critical periods, generalization estimation techniques, data

augmentation, and data pruning. Chapter 4 details the proposed method, including the adaptation of a key generalization estimation technique for systematically identifying critical periods. Chapter 5 presents the experimental setup, metrics, and a comprehensive analysis of the results across various architectures and datasets, including comparisons with state-of-the-art methods and discussions on computational benefits and Green AI. Chapter 6 explores the dynamics of critical learning periods in Transformer-like architectures. Finally, Chapter 7 concludes this work, summarizing the findings and outlining promising avenues for future research.

2 Background

This chapter establishes the fundamental concepts and notation employed throughout this dissertation. The reader finds here the necessary background on critical periods, generalization estimation techniques, data augmentation, and data pruning. The idea is to prepare the reader for a comprehensive understanding of our contributions.

2.1 Preliminaries

Assume X and Y a set of training samples and their respective class labels (i.e., categories). Let $\mathcal{F}(\cdot, \theta)$ be a neural network parameterized by a set of weights θ. From a random initialization θ^0, an iterative process (e.g., Stochastic Gradient Descent – SGD) updates θ^i towards a minimum of the loss function $\mathcal{L}$, where i indicates the i-th iteration of this process. To improve generalization of $\mathcal{F}$, previous works typically apply regularization and data augmentation mechanisms during the optimization iterative process (CUBUK et al., 2020; HENDRYCKS et al., 2022; KIM et al., 2025; ROBINE et al., 2025). Particularly, in this work, we focus on regularization through data augmentation techniques.

2.2 Critical Learning Periods

Critical learning periods encompass an important phenomenon involving deep learning. These periods refer to a deep learning phenomenon where early epochs play a decisive role in the success of many training recipes (GOLATKAR et al., 2019; ACHILLE et al., 2019). Such recipes include regularization, forms of increasing the capacity of the models and combining information from diverse sources (KLEINMAN et al., 2023; KLEINMAN et al., 2024). Importantly, understanding critical learning periods significantly boosts training efficiency and overall model performance (GOLATKAR et al., 2019; ACHILLE et al., 2019). Once these periods pass, persistently applying regularization yields diminishing returns, adding computational overhead without comparable benefits (KLEINMAN et al., 2023).

The training dynamics of neural networks reveal that early stages of learning play a decisive role in determining model performance (ACHILLE et al., 2019; GOLATKAR et al., 2019; MAINI et al., 2023). The seminal work by Kleinman et al. 2024 demonstrates that learning ability does not increase monotonically during training and that a memorization phase occurs in early epochs, where models retain most discriminative information. Furthermore, the authors show that critical periods occur during the memorization phase

and, after this, the model may start to weaken its ability to retain knowledge (forgetting phase).

2.3 Regularization

A central challenge within deep learning involves the development of algorithms with high predictive performance across both training data and novel inputs. Regularization involves any modification to a learning algorithm with the primary aim to minimize generalization error without impact on training error. Prominent strategies include Weight Decay (LOSHCHILOV; HUTTER, 2017), Dropout (WAGER et al., 2013), Early Stopping (YAO et al., 2007), Ensemble methods (DIETTERICH, 2000), and Data Augmentation (MIKOŁAJCZYK; GROCHOWSKI, 2018).

High generalization capacity necessitates superior predictive performance on both training distributions and novel samples. Regularization penalizes model complexity or smoothness to maintain model performance on out-of-sample data, even with finite training sets or sparse optimization steps. Modern deep learning expands this definition; regularization now encompasses any technique with the primary goal of increasing model generalization (TIAN; ZHANG, 2022). Among these methodologies, Data Augmentation stands out by inserting prior knowledge directly into the training pipeline via synthetic sample transformations. The next analysis focuses on this specific technique.

2.4 Data Augmentation

The current landscape of deep learning solution for cognitive tasks heavily relies on training models with vast amounts of data, often from web-scale sources (GRATTAFIORI et al., 2024). Modern architectures, such as Llama 3, underscore this reliance, affirming that high-quality, web-scale data forms the core of achieving strong results (GRATTAFIORI et al., 2024). In this direction, data augmentation becomes one of the most important training recipes. It reveals that, due to the stochastic nature of state-of-the-art techniques, many of them allow the creation of multiple samples from a single one (YUN et al., 2019). Thus, one increases both data quantity and diversity without a labor-intensive and costly labeling process. Taking 3 as an example, popular augmentations such as PixMix (HENDRYCKS et al., 2022), MixUp (ZHANG et al., 2018), and Cutout (DEVRIES; TAYLOR, 2017) apply transformations to the original sample with a given probability p. Note how some methods, like MixUp and CutMix, involve blending information from different classes, thereby altering the label (e.g., $y = \alpha \cdot \text{bird} + (1 - \alpha) \cdot \text{turtle}$), while others maintain the original label (e.g., $y = \text{bird}$).

From this perspective, a training process generates training datasets arbitrarily large

Figure 3 – Overview of various data augmentation techniques applied to a *clean image* of a bird. These methods transform the original sample to enhance dataset diversity and improve model generalization. Popular techniques include Random Flip (HE et al., 2016), Cutout (DEVRIES; TAYLOR, 2017), MixUp (ZHANG et al., 2018), CutMix (YUN et al., 2019), and PixMix (HENDRYCKS et al., 2022).

just using data augmentations k times per input. To simulate diverse viewing conditions and angles, it is important to prepare images by applying a variety of transformations through a data augmentation setup. Common transformations include:

- Random rotation: randomly rotate images within a range to make the model invariant to minor orientation changes of objects within the images.
- Width shift: shift images horizontally by up to a percentage of their total width to test the ability of the model to recognize objects that aren't perfectly centered.
- Height shift: similarly, shift images vertically by up to a percentage of their total height, challenging the model to maintain accuracy when objects appear at different vertical positions within the frame.
- Horizontal flip: randomly flip images horizontally, training the model to recognize mirrored versions of objects and thereby increasing its generalization capabilities.

These data augmentation techniques artificially expand the dataset and introduce a broader range of variability into the training data. This strategy aims to enhance the model generalization ability by exposing it to a wider spectrum of scenarios during training. Regardless of whether the increase in data quantity stems from collection or data augmentation, handling more data implies intensive computational cost, energy demand and financial cost.

From the previous techniques, we formalize the employing of data augmentation as follows. Let $T(\cdot)$ be a function that receives samples from X and modifies its content, (i.e.,

using the techniques in Figure 3) producing a new set of same size (i.e., $|T(X)| = |X|$). Typically, modern data augmentation methods incorporate stochastic elements enabling the creation of arbitrarily large datasets by applying T multiple times (CUBUK et al., 2020; HENDRYCKS et al., 2022; KIM et al., 2025). Formally, we augment the original dataset by applying T k times, denoting by $\{(T(X), Y)\}^k$. To simplify the notation, let $\mathcal{D} = (X, Y)$ represent the pair of samples and their respective labels. Thus, after performing data augmentation k times, it is possible to rewrite data augmentation as $\{\mathcal{D}\}^k$. By applying data augmentation techniques, we formalize the iterative process of updating θ as follows:

$$\theta^{i+1} = \theta^i - \eta \frac{1}{B} \sum_{b=1}^{B} \nabla \mathcal{L}(\{\mathcal{D}\}_b^k, \theta^i), \tag{2.1}$$

where B indicates the *batch size*, $\mathcal{D}_b^k$ is a *batch* of b samples from augmented data, $\nabla \mathcal{L}$[1] corresponds to the gradient of the loss function with respect to the parameters θ^i and η denote the update magnitude (learning rate).

2.5 Data Pruning

While data augmentation focuses on enriching the dataset, data pruning aims to reduce computational costs by selecting a smaller, representative subset of data, without sacrificing predictive performance. Existing methods typically fall into two categories. Importance-based approaches (HAN et al., 2023; XIA et al., 2024; CHOI et al., 2024) evaluate the relevance of individual data points. Optimization-based techniques aim to retain the core characteristics of the original dataset (MAHABADI; TRAJANOVSKI, 2023; ENGSTROM et al., 2024; LI et al., 2025). However, both approaches often involve complex and computationally expensive procedures. Interestingly, Okanovic et al. 2024 demonstrate that random pruning strategies, when well-conceived, match or even outperform many intricate methods, underscoring the potential of simpler, more scalable solutions. At each epoch, the process applies dynamic random data pruning by randomly selecting a percentage of the training dataset. Therefore, the training data changes with every epoch, ensuring the model sees different data points.

Formally, Let $\mathcal{D} = \{z_i\}_{i=1}^N = \{(x_i, y_i)\}_{i=1}^N$ denote a training dataset consisting of N samples, where x_i represents the input and y_i the target label. The standard goal when learning a deep learning model is to minimize the empirical risk over $\mathcal{D}$ by optimizing the parameters θ of a network $\mathcal{F}(x; \theta)$:

$$\min_{\theta} \mathcal{L}(\mathcal{D}; \theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(\mathcal{F}(x_i; \theta), y_i), \tag{2.2}$$

[1] It is possible to rewrite the gradient as follows: $\mathcal{L}(\{(Y_b, \mathcal{F}(T(X_b)))\}^k, \theta^i)$, where X_b and Y_b are data *batches* and respective labels.

Algorithm 1: Static Pruning Procedure

Input: Dataset $\mathcal{D}$, Scoring Function $H(\cdot)$, Threshold $\bar{H}$
Output: Static Subset $\mathcal{S}$

```
Initialize empty subset S ← ∅;
foreach sample z ∈ D do
    Assign score H(z);
    Compute pruning decision P(z; H) ← 𝟙(H(z) < H̄);
    if P(z; H) == 0 then
        Include sample in subset S ← S ∪ {z};
    end
end
return S;
```

where $\ell(\cdot)$ is a sample-wise loss function (e.g., Cross-Entropy). From this optimization perspective, data pruning aims to identify a subset $\mathcal{S} \subset \mathcal{D}$, $|\mathcal{S}| \ll N$ such that training $\mathcal{F}(\cdot;\theta)$ on $\mathcal{S}$ maintains predictive ability (i.e., accuracy) comparable to the full dataset $\mathcal{D}$, while significantly reducing the total training duration. We employ this training phase only when considering data pruning mechanisms. According to the literature (TONEVA et al., 2019; QIN et al., 2024), studies separate data pruning into two categories: static and dynamic pruning.

2.5.1 Static Pruning

In this setting, a data pruning algorithm assigns a scoring function $H(z)$ to each sample z prior to training. Then, a probability $P(z;H) \in \{0,1\}$ determines pruning decisions. For example, Toneva et al. 2019 define the pruning rule as:

$$P(z;H) = \mathbb{1}(H(z) < \bar{H}), \tag{2.3}$$

where $\bar{H}$ is a pre-defined threshold and $\mathbb{1}(\cdot)$ is the indicator function. The method constructs a static subset $\mathcal{S}$ discarding samples where $P(z;H) = 1$ before optimization (i.e., Eq. 2.2) begins.

Algorithm 1 summarizes the static pruning mechanism. This strategy involve a single evaluation of the dataset prior to the optimization phase. By applying a scoring function to assess sample utility, the method yields a stable subset for all training epochs. This approach ensures minimal computational overhead since the reduction occurs before the learning process begins. Consequently, the technique offers a straightforward path to efficiency by early removing redundant information.

2.5.2 Dynamic Pruning

Unlike static approaches, dynamic pruning selects data adaptively throughout the training process. Here, the importance score H_t varies with the training step t, reflecting

Algorithm 2: Dynamic Pruning Procedure

Input: Full Dataset $\mathcal{D}$, Total Epochs T, Initial Parameters θ^0
Output: Final Model θ^T

for $t \leftarrow 1$ **to** T **do**
  Initialize empty subset $\mathcal{S}_t \leftarrow \emptyset$;
  foreach *sample* $z \in \mathcal{D}$ **do**
    Compute importance score $H_t(z)$ via current parameters θ^{t-1};
    Determine selection probability $P_t(z) \leftarrow P(z; H_t)$;
    if *Sample* z *satisfies selection criteria via* $P_t(z)$ **then**
      Include sample in active subset $\mathcal{S}_t \leftarrow \mathcal{S}_t \cup \{z\}$;
    end
  end
  Update parameters θ^t by training on subset $\mathcal{S}_t$;
end
return θ^T;

the temporal status of the learning process. The probability becomes step-dependent, $P_t(z) = P(z; H_t)$, forming a dynamically evolving subset $\mathcal{S}_t$ at each epoch. Crucially, dynamic pruning leverages the full dataset $\mathcal{D}$ (i.e., subsets) over the course of training and results in a significantly lower gradient expectation bias compared to static methods.

Algorithm 2 formalizes the adaptive nature of dynamic pruning. This approach selects samples throughout the optimization phase. By re-evaluating importance scores at every epoch, the framework allows for a varying training subset. This continuous adjustment minimizes the gradient expectation bias while maximizing information gain. Consequently, the approach maintains a more representative data distribution relative to static selection.

2.6 Generalization Estimation

Estimating how neural networks generalize to unseen data early in the training process confirms crucial for both practical applications and for advancing the theoretical understanding of deep models (BALLESTER et al., 2024). In this direction, various methods exist to assess generalization during training. For example, Sankararaman et al. 2020 estimate training convergence using the gradient confusion among batches of samples during the SGD updates. Gradient confusion quantifies the alignment between the gradients of two distinct batches of data during a specific training epoch. We compute the cosine similarity between gradients through the following mathematical formula:

$$CG(A, B) = \frac{A \times B}{\|A\|\|B\|}, \; CG(A, B) \in [-1, 1],$$

where A and B represent the gradients of two batches of data. A value close to 1 indicates high alignment (low confusion), while a value near -1 reflects high confusion between the

gradients. Similarly, Chen et al. 2023 estimate training dynamics according to the stability of gradient directions across batches and parameter updates, a metric they term Gradient Predictiveness. It quantifies the alignment between the gradient of a batch at epoch i and the gradient of the same batch at epoch $i-1$. This metric evaluates the temporal consistency of gradient updates, offering insight into the stability of the training process over consecutive epochs. Mathematically, its definition is the cosine similarity between the gradient vectors in terms of:

$$GP(G_t, G_{t-1}) = \frac{G_t \times G_{t-1}}{\|G_t\|\|G_{t-1}\|}, \quad GP(G_t, G_{t-1}) \in [-1, 1],$$

where G_i and G_{i-1} represent the gradients of a given batch at epochs i and $i-1$, respectively. A value close to 1 indicates strong alignment (high predictiveness), whereas a value near -1 suggests substantial variation in the gradient direction (low predictiveness).

One method that stands out for its efficiency and direct relevance to generalization is Layer Rotation (CARBONNELLE; VLEESCHOUWER, 2019). Since it becomes a key component of our method for systematically identifying critical periods. Given its importance and alignment within our method, we explain Layer Rotation in more details in Chapter 4.

3 Related Work

This chapter reviews key works related to the scope of this dissertation, covering foundational research and recent advancements that inform our approach.

3.1 Critical Learning Periods

Golatkar et al. 2019 address the question of when to apply regularization during training, suggesting that the early epochs prove decisive for the learning outcome of a deep neural network. The authors test the hypothesis that applying regularization at different epochs yields different outcomes. Their findings show that applying weight decay or data augmentation beyond the initial transient of training does not improve generalization, as this transient is decisive for asymptotic performance. They also observe that regularization applied only during the final phases of convergence brings no generalization improvements. A key discovery is that the effect of regularization is most pronounced during a short, critical period. This led them to conclude that the analysis of regularization in deep networks should focus on the transient, rather than asymptotics. To support these findings, the authors train standard DNN architectures (ResNet-18/All-CNN) on the CIFAR-10/CIFAR-100 datasets.

In a related line of inquiry, Kleinman et al. 2024 investigate the emergence of critical learning periods in deep networks. They demonstrate that the ability of a neural network to integrate information from diverse sources hinges critically on being exposed to properly correlated signals during the early phases of training. This phenomenon, also known as a critical learning period in biological systems, shows that interfering with the learning process during this initial stage permanently impairs a development. The authors argue that these critical periods arise from the complex and unstable early transient dynamics, thus, they are decisive for the final performance of a trained system and its learned representations. This work challenges the view that early learning dynamics in neural networks are simple, showing that even deep linear networks exhibit critical learning periods for multi-source integration, while shallow networks do not. To understand how internal representations change, the authors introduce a new measure called *source sensitivity* to track the inhibition and integration of sources during training. They find that architectures trained with cross-sensor reconstruction objectives are remarkably more resilient to critical periods.

3.2 Data Augmentation

Data augmentation is a fundamental technique for improving model generalization and robustness in deep learning by artificially expanding the dataset (CUBUK et al., 2020; DEVRIES; TAYLOR, 2017). The evolution of this strategy goes beyond simple transformations, with recent research exploring more complex and effective methods to enhance model performance. (HENDRYCKS et al., 2022) The following works highlight some of these modern approaches.

Hendrycks et al. 2022 propose PixMix, a data augmentation technique that improves various machine learning safety measures beyond accuracy. This approach introduces new complexity into the training procedure by integrating diverse patterns from fractals and feature visualizations into the training set. The PixMix method comprises two main components: a set of structurally complex pictures (*Pix*) and a pipeline for augmenting clean training pictures (*Mix*). The authors evaluate PixMix on extensions of CIFAR-10, CIFAR-100, and ImageNet-1K for various safety tasks while also evaluating on the standard versions of these datasets to check original performance. They find that PixMix is the first method to provide substantial improvements over the baseline on all existing safety metrics, obtaining state-of-the-art performance in nearly all settings.

Similar to Hendrycks et al. 2022, Zhang et al. 2018 propose MixUp, a learning principle to alleviate undesirable behaviors in large deep neural networks such as memorization and sensitivity to adversarial examples. The core of the MixUp method is to train a neural network on convex combinations of pairs of examples and their labels. The idea behind this process is to regularize the network to favor simple linear behavior between training examples. Extensive experiments on ImageNet-2012, CIFAR-10, CIFAR-100, Google commands, and UCI datasets show that MixUp improves the generalization of state-of-the-art neural network architectures. The authors also find that MixUp reduces the memorization of corrupt labels, increases robustness to adversarial examples, and stabilizes the training of generative adversarial networks.

To address the information loss and inefficiency issues of regional dropout methods, Yun et al. 2019 introduce an innovative augmentation strategy with CutMix. These methods remove informative pixels with black patches or noise. The CutMix approach cuts and pastes patches among training images, mixing the ground truth labels proportionally to the area of the patches. The authors demonstrate that by making efficient use of training pixels and retaining the regularization effect of regional dropout, CutMix consistently outperforms state-of-the-art augmentation strategies on CIFAR and ImageNet datasets. Furthermore, a CutMix-trained ImageNet model results in consistent accuracy gains when used as a pretrained model for Pascal detection and MS-COCO image captioning benchmarks. Their work also shows that CutMix improves model robustness against input corruptions and out-of-distribution detection performances.

In a related effort, DeVries et al. 2017 propose Cutout, a simple regularization technique motivated by the fact that powerful CNNs are often susceptible to overfitting and require proper regularization to generalize well. Cutout randomly masks out square regions of input during training to improve the robustness and overall performance of convolutional neural networks. The authors demonstrate that this method is not only easy to implement but also works in conjunction with existing forms of data augmentation and other regularizers to further improve model performance. They evaluate this method on state-of-the-art architectures using the CIFAR-10, CIFAR-100, and SVHN datasets, achieving new state-of-the-art results.

3.3 Data Pruning

Data pruning selects a smaller, informative subset of a large dataset and emerges as a key strategy to address the computational challenges of training neural networks (QIN et al., 2024). The following works present a variety of approaches, from universal methods that adapt to different selection ratios to innovative techniques tailored for specific architectures and tasks. Overall, all techniques aim to reduce costs without compromising performance.

In a related effort, Choi et al. 2024 introduce a universal and efficient data subset selection method, Best Window Selection (BWS), to address the challenge of finding a single approach that consistently achieves competitive performance across a broad range of selection ratios. The authors note that existing methods tend to specialize in either high or low selection ratio regimes, lacking a universal approach that approximates full-dataset training for large datasets. BWS works by proposing a method to choose the best window subset from samples ordered based on their difficulty scores, offering flexibility through the choice of window intervals. To efficiently select this subset, BWS evaluates its quality using kernel ridge regression. Experimental results on CIFAR-10/100 and ImageNet demonstrate the superior performance of BWS compared to other baselines across a broad range of selection ratios, in scenarios involving both training from random initialization and fine-tuning of pre-trained models.

While some approaches favor complex strategies, Okanovic et al. 2024 propose Repeated Sampling of Random Subsets (RSRS), a simple yet powerful random sampling strategy to address the challenge of reducing *time-to-accuracy* in deep neural network training. Many existing data pruning and distillation methods, despite their complex designs, come with additional computational costs for subset selection. These methods even underperform random sampling in high data compression regimes. In this context, RSRS operates by randomly sampling a subset of the training data for each epoch of model training. The authors evaluate RSRS against thirty state-of-the-art data pruning and data distillation methods across four datasets, including ImageNet. Overall, the results confirm

that RSRS significantly reduces time-to-accuracy. For example, RSRS yields accuracy improvements up to 29% and a runtime reduction of 7× when training on ImageNet in a high-compression regime. Importantly, RSRS establishes as a competitive baseline for future data selection or distillation techniques aimed at efficient training.

However, Yuan et al. 2025 introduce an *Instance-dependent Early Stopping* (IES) method to improve training efficiency by addressing redundant computations on instances the model already masters. Conventional early stopping applies the same criterion to all instances, regardless of their individual learning status. The IES method operates on the principle that training on an instance should stop once a model completely learns it. The authors consider an instance mastered if the second-order differences of its loss value remain within a small range around zero. Such stability provides a more consistent measure of learning status compared to using the loss value directly. Excluding fully learned instances from backpropagation increases gradient norms, thereby accelerating the decrease of the training loss and speeding up the training process. Extensive experiments demonstrate that IES reduces backpropagation instances by 10-50% while maintaining or even slightly improving test accuracy and transfer learning performance.

Qin et al. 2024 introduce *InfoBatch*, a framework designed to achieve lossless training acceleration through unbiased dynamic data pruning. The InfoBatch approach selectively prunes a portion of less informative samples based on their loss distribution, then rescales the gradients of the remaining samples to approximate the original gradient. As a plug-and-play, architecture-agnostic framework, InfoBatch consistently achieves lossless training results across various tasks, including classification, semantic segmentation, and instruction fine-tuning. The authors demonstrate that InfoBatch losslessly reduces overall cost by 40% on datasets like CIFAR-10/100 and ImageNet-1K. For LLaMA instruction fine-tuning, combining InfoBatch with a coreset selection method yields a 10× acceleration. These results encourage further exploration into the data efficiency of large model training.

Additionally, Qin et al. 2024 highlight that exposing the model to the full dataset during the final epochs, after applying data pruning, improves accuracy. This exposure allows the model to learn from data points that face exclusion by pruning. To achieve this, they introduce the Annealing parameter $\delta \in (0, 1)$. This parameter controls when the model reverts to the full dataset: pruning only during the first $\delta \cdot N$ epochs, where N is the total number of epochs in training. After that, the authors train the model on the complete dataset for the remaining epochs. Therefore, higher values of δ lead to a greater reduction in training costs.

In contrast to these state-of-the-art methods, our framework formalizes Critical Learning Periods within the pruning domain. We demonstrate that the timing of data removal is just as critical as the selection criteria itself. Furthermore, we address a fundamental limitation in existing data pruning strategies: the uniform application of

reduction heuristics that neglect the decisive influence of early training epochs.

3.4 Generalization Estimation

Generalization estimation is crucial for both practical applications and for advancing the theoretical understanding of deep models (SANKARARAMAN et al., 2020; CHEN et al., 2023). The literature explores various metrics to quantify training dynamics, with the following works offering distinct perspectives on how to assess the learning process of a model and its potential for generalization.

Sankararaman et al. 2020 study how neural network architecture affects the speed of training, introducing a concept called gradient confusion to formally analyze this relationship. The authors explain that high gradient confusion, where stochastic gradients from different data samples may correlated negatively, slows down convergence, while low confusion allows training to proceed quickly. Through theoretical and experimental results, they demonstrate that for popular initialization techniques, increasing network width leads to lower gradient confusion and thus faster training. Conversely, increasing network depth has the opposite effect, but they note that alternate initialization techniques or networks using both batch normalization and skip connections help reduce the training burden of very deep networks.

Building on the understanding of training dynamics, Chen et al. 2023 systematically explore the layer-wise convergence rate of deep neural networks, motivated by the complex interaction between layers that makes the learning process of a particular layer poorly understood. They demonstrate a phenomenon called *Layer Convergence Bias*, where shallower layers tend to converge faster than deeper layers. The authors uncover the fundamental reason behind this phenomenon: the flatter local minima of shallower layers make their gradients more stable and predictive, allowing faster training. Another finding is that shallower layers tend to learn the low-frequency components of the target function, while deeper layers usually learn the high-frequency components. This is consistent with the discovery that deep neural networks learn lower frequency objects faster.

4 Proposed Method

To grasp the intriguing generalization properties present in deep neural networks, it is crucial to identify numerical indicators of generalization performance that remain applicable across diverse training settings. In this context, Carbonnelle et al. 2019 propose a groundbreaking approach to understanding and improving neural network generalization, named *Layer Rotation*. This approach focuses on tracking the evolution of the cosine distance between each weight vector of each layer and its initial state throughout the training process. Following Mason-Williams et al. 2024, instead of computing the cosine similarity between each weight of a given layer (as originally suggested by Carbonnelle et al. 2019), we concatenate all the weights composing the model $\mathcal{F}$ and linearize them to form a single vector. According to their work (MASON-WILLIAMS; DAHLQVIST, 2024), this process enables a representation of the neural network parameters. For ease of exposition, we keep indicating the single vector representing all weights of $\mathcal{F}$ as θ^0 (random initialization) and θ^i (with $i > 0$).

Given the previous definition, the cosine distance between θ^0 and θ^i defines layer rotation at training step i. Thus, we estimate the layer rotation in terms of

$$d_{cos} = 1 - \frac{\theta^{\mathbf{0}} \cdot \theta^{\mathbf{i}}}{\|\theta^{\mathbf{0}}\| \|\theta^{\mathbf{i}}\|}. \tag{4.1}$$

Through a comprehensive suite of experiments encompassing a broad spectrum of datasets, network architectures, and training regimes, we uncover a consistent pattern: *larger layer rotations (i.e. as cosine distance between the final and initial weights increases) reliably predict enhanced generalization performance.*

In order to visualize layer rotation evolution during training, we track the cosine distance between the current weight vector of each layer and its initial state across various training steps. Upon analyzing these curves on a validation set, we notice a characteristic shape pattern emerging throughout training process. To systematically identify critical periods within this evolution, we adopt an approach that performs linear regression over a window of 5 epochs (w). Figure 4 illustrates this process across 50 initial epochs. The choice of monitoring 5 epochs proves itself effective in our experiments, emerging as the smallest window size that still allows capturing significant changes in layer behavior. A smaller window would make the analysis too vulnerable to minor fluctuations.

We scale the number of epochs and cosine distance from 0 to their respective maximum values. This way, we graphically calculate the learning variation during training by the angle α that indicates the rate of change within this window. Then, we determine its value by the linear regression of the normalized data and the arctangent calculation

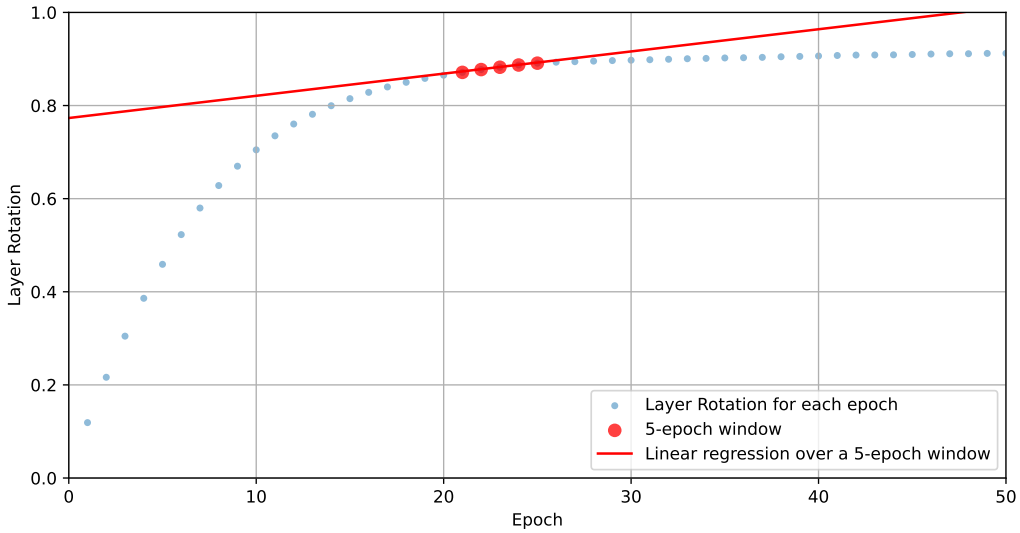


Figure 4 – Linear regression analysis over a 5 epochs window (red points) in ResNet44 training on CIFAR-10, within the initial 50 epochs, showcasing a regression angle of 43.63 degrees. The analysis reveals critical learning phases and trends, guiding optimization efforts by pinpointing pivotal changes.

of the regression coefficient m in degrees. To accomplish this, given a set of data points $\{(u_1, v_1), (u_2, v_2), \ldots, (u_w, v_w)\}$, where u_i represents the epoch (independent variable) and v_i the cosine distance between θ^i and θ^0 (dependent variable) – the Layer Rotation, the slope m of the regression line that best fits these data points (in the least squares sense) is:

$$m = \frac{\sum_{i=1}^{w}(u_i - \bar{u})(v_i - \bar{v})}{\sum_{i=1}^{w}(u_i - \bar{u})^2}. \tag{4.2}$$

We therefore present the formula for calculating the angle as follows:

$$\alpha = \frac{180}{\pi} \arctan(m). \tag{4.3}$$

In our initial experiments, we observe that an angle of 45° marks a pivotal moment in the training of neural networks, where the shift from rapid learning to careful refinement and optimization occurs. Therefore, we use this value throughout our work.

Algorithm 3 summarizes our strategy. The core mechanism relies on a dynamic conditional check at each epoch: calculating the Layer Rotation d_{cos} to assess model stability. Overall, during the initial phase, when the indicator suggests the model remains within the critical learning period, the algorithm strictly enforces training on the full dataset $\mathcal{D}$, protecting the formation of essential representations. Once the metric surpasses the critical boundary, our strategy transitions to a resource-efficient mode, activating a data pruning technique to reduce the training subset $\mathcal{S}_t$.

Algorithm 3: Overview of the Proposed Method

Input: Model $\mathcal{F}$, Total Epochs T, Dataset $\mathcal{D}$
Output: Trained Model $\mathcal{F}$

Initialize $\mathcal{F}$ with parameters θ^0
Initialize active training data $\mathcal{D}_{active} \leftarrow \mathcal{D}$
for $t \leftarrow 1$ *to* T **do**
    Train $\mathcal{F}$ on $\mathcal{D}_{active}$
    Update parameters to θ^t (e.g., any optimization strategy) using Eq. 2.2
    Calculate Layer Rotation $d_{cos}(\theta^0, \theta^t)$ using Eq. 4.1
    if *Critical Period Identified (based on* d_{cos}*)* **then**
        ▷ Switch to resource-efficient mode
        Apply data pruning to obtain subset $\mathcal{S}_t \subset \mathcal{D}$
        $\mathcal{D}_{active} \leftarrow \mathcal{S}_t$
    else
        ▷ Critical Learning Period: Maintain full data
        $\mathcal{D}_{active} \leftarrow \mathcal{D}$
    end
end
return Trained Model $\mathcal{F}$

Beyond the main analysis of Layer Rotation, we investigate alternative measures, such as Gradient Confusion and Gradient Predictiveness. Appendix A contains the complete results for these metrics. Furthermore, they remain compatible with both Algorithm 1 and Algorithm 2, that is, it is possible to apply any data pruning method after identifying critical learning periods using any of these metrics.

5 Experiments

5.1 Experimental Setup

In our experiments, we use a standard training protocol of 200 epochs and SGD optimizer with learning rate that starts at 0.01, and divide by 10 at epochs 100 and 150. Furthermore, our data transforms randomly before each epoch by a combination of horizontal flips, crops, rotations and translations. We apply this basic transformation throughout the training process, regardless of any reduction in other augmentation factors. Unless stated otherwise, our data augmentation consists of repeating each sample k times, with $k = 3$ (formalism given in Section 2.4). By doing this repetition step before the random transformations, we ensure each copy is slightly different from the original. This is important because we want to simulate the effect of having more samples, not just simple copies. We employ these specific augmentation strategies to ensure parity with standard training pipelines.

Regarding the models and datasets, we use the popular Residual networks (HE et al., 2016), at different depths, and the CIFAR-10/100 benchmarks (KRIZHEVSKY et al., 2009a; KRIZHEVSKY et al., 2009b). Overall, we apply these experimental settings (model $\times$ datasets) for most experiments because they are common practices in the context of critical period and data pruning (QIN et al., 2024; GOLATKAR et al., 2019; KLEINMAN et al., 2023). However, to confirm the effectiveness of our method, we also evaluate it on large-scale datasets, including EuroSAT (HELBER et al., 2018), Tiny ImageNet (LE; YANG, 2015), and ImageNet30 (HENDRYCKS et al., 2019).

Throughout experiments and discussions, the term baseline refers to the model without removing training recipes. In other words, it means the model training on standard practices without any knowledge and intervention of critical periods.

5.2 Metrics and Evaluation

To evaluate the effectiveness of our method in terms of improvements in computational cost and generalization, we introduce two key metrics:

- Normalized training cost: this metric quantifies the computational demand of training relative to the baseline (training with initial $k = 3$ – see Equation 2.1 – during all epochs). It normalizes the cost by combining the number of epochs and the

number of data points used per SGD updates. Overall, this metric accounts for dynamic changes in dataset size during the training process, particularly during critical periods.

- Accuracy delta: this metric measures the variation in model accuracy compared to the baseline model. It enables assessing the trade-off between computational efficiency and model predictive performance when removing training recipes.

5.3 Enhancing Generalization Through Repeated Augmentation

We start our analysis by illustrating the advantages of generating arbitrarily large datasets by applying the same data augmentation k times per sample. As we mentioned before, this is possible due to the stochastic nature of modern augmentation strategies. Specifically, because they employ transformations (i.e., rotation or crop) with a given probability p. To this end, we increase the number of training samples by three times ($k = 3$ in Equation 2.1) and compare the accuracy when using the dataset with data augmentation but without changing its size (i.e., $k = 1$). For a fair comparison, we consider the same initialization.

On the ResNet32 architecture, we observe an improvement of roughly 1 percentage point (pp) while using the increased dataset ($k = 3$) and a similar gain with a deeper, high-capacity architecture (ResNet86[1]).

From these findings, we confirm that an effective training recipe for improving generalization is simply to expand the dataset size by repeating the same data augmentation k times. Additionally, we highlight the following key observations. First, although these improvements may seem small, modern and complex data augmentation methods achieve similar gains (ZHANG et al., 2018; ZHONG et al., 2020). Second, despite improving generalization, the training time increases proportionally; for example, moving from $k = 1$ to $k = 3$ increases the training time by roughly three times. Most importantly, this setting becomes a potential candidate for exploring the practical benefits of our method in discovering critical periods. In particular, we begin training a model on the expanded version of a dataset ($k = 3$) and then reduce the dataset to its original size ($k = 1$), or even use smaller versions ($k < 1$ – data pruning), after identifying the critical period. Therefore, adapting the training size through k enables leveraging the best of both worlds: *higher generalization and lower training time.*

[1] Here, we avoid using the popular ResNet56 and ResNet110 to prevent any bias in our subsequent analysis.

5.4 Revisiting Critical Periods

In this experiment, we revise the existence of critical periods. However, in contrast to previous works (KLEINMAN et al., 2024; KLEINMAN et al., 2023; ACHILLE et al., 2019), we analyze it from the lens of potentials i^*, taking into account the compromise between accuracy and computational cost. Specifically, our main objective is to identify the end of the critical period, i.e., an early epoch i^* where we reduce training recipes until the training is complete and final accuracy is sufficiently close to the baseline accuracy. To achieve this, we create oracle models that reveal every possible accuracy outcome after reducing the augmentation factor k from 3 to 1 at epoch i. It is worth mentioning that, after eliminating data augmentation at epoch i, the training continues until completing 200 epochs. Therefore, each oracle model uses $k = 3$ for the initial i epochs and $k = 1$ for the remaining $200 - i$ epochs.

The aforementioned process produces 200 oracle models and to mitigate the impact of outliers, we execute each experiment three times. Figure 5 shows the accuracy of each model at the end of the training phase. The figure reinforces the existence of critical periods early on training and supports that, after pinpointing i^* (i.e. the end of the critical period), reducing augmentation is effective in terms of accuracy and computational cost. Following this reasoning, epoch 24 marks the potential end of the critical period, as it provides an interesting balance between accuracy and computational cost. It is important to note that better-performing models only emerge by epoch 60, but the accuracy gain is not significant enough to justify the additional computational cost.

Even by taking advantage of checkpoints to skip the first repeated initial epochs of every training session, this experiment is very resource-intensive. For this reason, we choose CIFAR-10 and ResNet32, as both are computationally light while remaining sufficiently challenging and capable of delivering competitive results for this task (HE et al., 2016; DENG et al., 2020). This combination represents the standard benchmark architecture within the field. We further explore multiple network depths to analyze architectural influence. Our findings reveal that depth variations directly change the temporal boundary of the critical period.

From Figure 5, the epoch i^* is explicit, however, this is the epoch we wish to identify without completing all these multiple models, allowing the decision to stop training recipes at that point in a single run (i.e., on-the-fly during a single training phase). This is what our method aims to achieve. Therefore, we now turn our attention towards automatically and systematically identifying i^*.

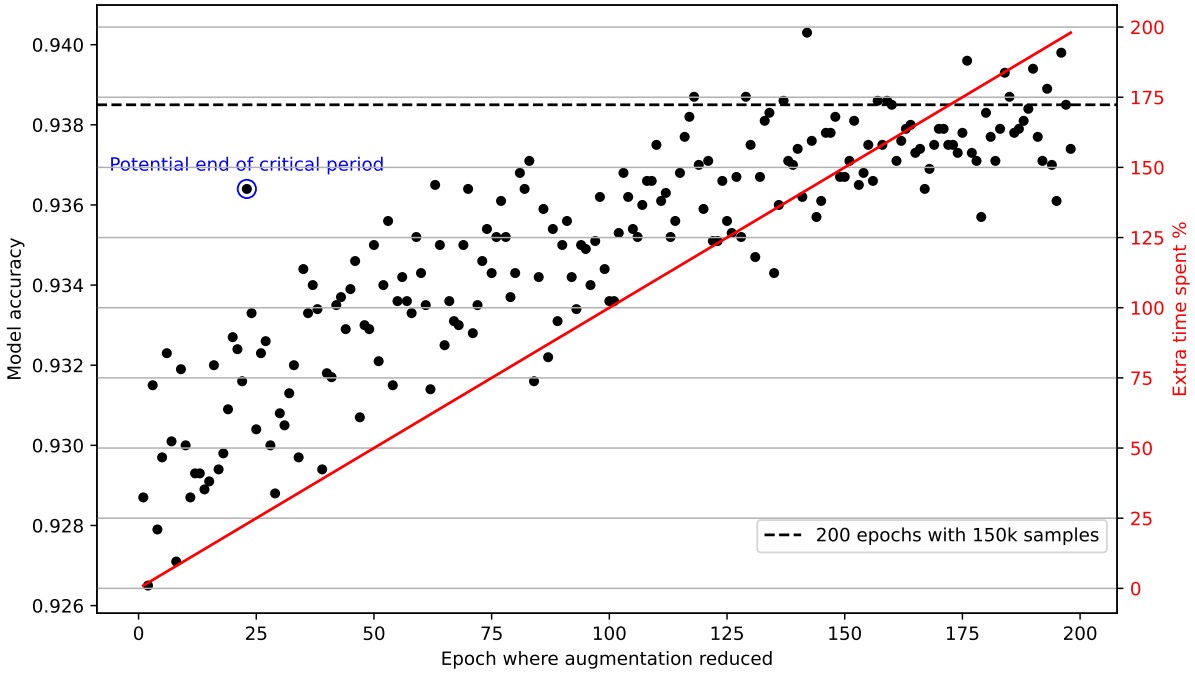


Figure 5 – Critical period in ResNet32 training on CIFAR-10. Each point indicates a model with a complete training cycle of 200 epochs. The figure shows the accuracy trends when data augmentation reduces from 150k to 50k samples per epoch at different epochs. The red line indicates the extra time cost.

5.5 Effectiveness of Generalization Estimators

Due to the nature of critical periods – epochs where training recipes promote significant impact on generalization (GOLATKAR et al., 2019; ACHILLE et al., 2019) – we argue that generalization estimation metrics successfully identify i^*. Specifically, these metrics enable estimating the epoch where generalization begins to decline by analyzing their behavior among a range of epochs during training. Informally speaking, we believe these metrics identify the circled point in Figure 5.

Knowing the effective i^* beforehand from the previous experiment enables validating our argument to use generalization estimation metrics. In addition to Layer Rotation, we explore the following metrics: Gradient Confusion (SANKARARAMAN et al., 2020) and Gradient Predictiveness (CHEN et al., 2023).

Regarding these two metrics, we observe notable drawbacks. For example, Gradient Confusion is capable of accurately identifying critical periods (i.e., identify the epoch 24 in Figure 5), but its computational cost is as intensive as the cost saved by reducing training recipes, making it impractical for real-time applications where reducing training time is a key objective. In contrast, Gradient Predictiveness identifies the critical period far from the effective point. Particularly, this metrics shows no correlation between its values and model accuracy on a validation set.

Regarding Layer Rotation, we observe that as epochs progress, the cosine distance

between the weights of each epoch and the initial weights increases. However, we notice that, at a certain point, the growth of this distance begins to diminish. Carbonnelle et al. 2019 state that Layer Rotation achieves a network-independent optimum when the cosine distance of all layers reaches 1. However, our practical observations contradict this, revealing that distance values significantly fluctuate with the architecture. Consequently, the pure application of this metric turns out to be architecture-dependent, making the Layer Rotation value arbitrary and unique to each architecture. Therefore, it becomes necessary to implement the learning variation during a window of epochs we introduce in Equations 4.2 and 4.3. These changes allow analyzing the slope of the line generated by the linear regression of a 5-epoch window, evidencing the reduction in generalization over epochs (i.e., temporally). From these experiments, we identify epoch 24 as the turning point, at which the angle α becomes less than 45^{o}, indicating the transition of the curve to a smoother phase. The comparison with the oracle in Figure 5 highlights this point as having significant potential for the critical period. Thereby, we confirm the potential of Layer Rotation in discovering the end of critical periods.

Besides accurately identifying the critical period, Layer Rotation is a quite efficient metric that demands negligible computational resources, as its calculations are simple and always based on comparing the current weights of the epoch with the initial ones (see Equation 4.1). Therefore, we employ Layer Rotation as the main metric implemented in our systematic method.

5.6 Comparison with State-of-the-Art Data Pruning

Existing data pruning methods provide positive results in reducing computational costs during the training of deep models (QIN et al., 2024; CHOI et al., 2024). However, to the best of our knowledge, no method currently takes into account the critical period phenomenon. In the next experiments, we investigate the behavior of applying data pruning *only after* our method identifies the conclusion of critical periods and compare the results with state-of-the-art data pruning methods following their original experimental setup.

5.6.1 Comparison with IES Method

To demonstrate the efficacy of our strategy, we leverage the Instance-dependent Early Stopping (IES) (YUAN et al., 2025) technique as a testbed. By integrating our critical period identification with IES, we isolate the impact of delaying data removal until achieving the learning stability. This combination allows us to assess whether respecting the critical period mitigate the risks associated with early, heuristic-based pruning while preserving the efficiency gains inherent to the IES approach.

Table 1 – Accuracy comparison (Acc) for IES (YUAN et al., 2025) (i.e., using IES as the data pruning technique in Line 9 of Algorithm 3). Original denotes the standard IES method, Ours refers to IES applied only after the critical period, and Baseline represents training on the full dataset without pruning. Bold and underline indicate the best and second-best results, respectively.

	Dataset	CIFAR-10			CIFAR-100		
	Architecture	ResNet18	ResNet50	VGG16	ResNet34	ResNet101	DenseNet121
IES	Original	0.9469	0.9464	0.9312	0.7747	<u>0.7799</u>	<u>0.7912</u>
	Ours	<u>0.9489</u>	<u>0.9474</u>	**0.9331**	**0.7754**	0.7788	**0.7930**
	Baseline	**0.9504**	**0.9488**	<u>0.9329</u>	<u>0.7752</u>	**0.7815**	0.7907

To this end, we evaluate the impact of combining our critical period strategy with the IES technique (YUAN et al., 2025) across six distinct architectures. Table 1 summarizes the results and demonstrates that applying pruning only after the critical period (*Ours*) allows maintaining accuracy close to or even higher than that obtained with pruning throughout the entire training (*Original*, IES (YUAN et al., 2025)). For example, for the ResNet18 architecture on CIFAR-10, our approach achieves 94.89% accuracy, surpassing the *Original* IES (YUAN et al., 2025) scenario (94.69%) and coming very close to the *Baseline* (95.04%). Notably, in cases like VGG16 and DenseNet121, our approach even surpasses the baseline, suggesting better data allocation by providing more samples at times of higher learning rates. Such results reinforce that initial phases represent the most delicate periods for training (GOLATKAR et al., 2019).

Figure 1 highlights the divergent behaviors in late-stage training (epochs 160–200), where our method successfully avoids the re-introduction of previously pruned samples. The original IES brings samples back into the training set to recover lost accuracy, a symptom of pruning too early. In contrast, our approach maintains a consistently compact dataset throughout this phase, avoiding this inefficient re-learning behavior precisely because it respects the critical period before initiating reduction. Note that IES re-introduces samples in late epochs (e.g., 160-200) to recover predictive performance. The table (top-right) reports the **mean accuracy drop** compared to the full baseline (lower value is better) on stardard benchamarks in data pruning. Our method achieves near-lossless accuracy (**-0.09** vs. -0.26 for IES on CIFAR-10) by preserving data during the early phase (i.e., the critical learning period [16]). We observe the same behavior when applying our method on other state-of-the-art data pruning techniques, such as InfoBatch (QIN et al., 2024) and random pruning (OKANOVIC et al., 2024), as follows.

5.6.2 Comparison with InfoBatch Method

We perform a similar analysis with the InfoBatch technique (QIN et al., 2024), testing on ResNet18, ResNet50, and ResNet101 architectures. The results in Table 2

Table 2 – Accuracy comparison (Acc) for InfoBatch (QIN et al., 2024) (i.e., using InfoBatch as the data pruning technique in Line 9 of Algorithm 3). Original denotes the standard InfoBatch method, Ours refers to InfoBatch applied only after the critical period, and Baseline represents training on the full dataset without pruning. Bold and underline indicate the best and second-best results, respectively.

	Dataset	CIFAR-10			CIFAR-100		
	Architecture	ResNet18	ResNet50	ResNet101	ResNet18	ResNet50	ResNet101
Info Batch	Original	0.9484	0.9400	0.9438	0.7796	<u>0.7701</u>	0.7788
	Ours	<u>0.9501</u>	<u>0.9404</u>	<u>0.9451</u>	<u>0.7830</u>	0.7687	<u>0.7830</u>
	Baseline	**0.9534**	**0.9495**	**0.9513**	**0.7866**	**0.7777**	**0.7941**

substantiate our earlier conclusions regarding IES. Our approach consistently positions itself as an effective balance point, with intermediate predictive performance between the *Original*, InfoBatch (QIN et al., 2024), and the *Baseline*. This highlights that respecting the critical period is fundamental to preserving the predictive capacity when applying data pruning techniques. Specifically, our method consistently outperforms the original InfoBatch configuration across all tested scenarios. For example, on CIFAR-10 on ResNet18, our approach increases accuracy to **95.01%**, surpassing the original InfoBatch result of 94.84% and narrowing the gap to the baseline (95.34%). Similarly, on CIFAR-100, we observe consistent gains, such as improving ResNet18 accuracy from 77.96% to **78.30%**. Our approach consistently positions itself as an effective balance point, delivering intermediate predictive performance between the *Original*, InfoBatch (QIN et al., 2024), and the *Baseline*. This highlights that respecting the critical period is fundamental to preserve the predictive capacity of the model when applying data pruning techniques.

5.6.3 Comparison with other Data Pruning Methods

Apart from the comparison with IES and InfoBatch, we also assess our method with other state-of-the-art methods. Table 3 introduces the results and confirms that our method outperforms all static pruning techniques across both datasets and pruning ratios, consistently demonstrating lower accuracy degradation. Against dynamic methods, our approach remains highly competitive, matching the gain of accuracy of the state-of-the-art InfoBatch method (QIN et al., 2024). Specifically, on CIFAR-10 with a 50% pruning ratio, our method incurs a minimal accuracy drop of just 0.3 percentage points (pp). Moreover, on CIFAR-100 with a 30% pruning ratio, our approach yields a notable accuracy gain of 0.2 pp, highlighting its robustness and consistency across methods and datasets. These results underscore the critical role of timing in data pruning: by delaying reduction mechanisms until after the critical period concludes, we enable the model to leverage the full dataset during its most effective learning phase, thereby optimizing model performance while minimizing trade-offs.

Table 3 – Comparison of accuracy delta (ΔAcc) with state-of-the-art data pruning methods. We report results on **ResNet18** as it represents the most common architecture in data pruning literature. Bold, underline, ↑ and ↓ mean the best results, second-best results, accuracy improvement and accuracy decrease, respectively.

	Dataset	CIFAR-10			CIFAR-100		
	Pruning Ratio %	30	50	70	30	50	70
Static	Random	↓1.0	↓2.3	↓5.4	↓4.4	↓6.1	↓8.5
	CD (AGARWAL et al., 2020)	↓0.6	↓1.3	↓4.8	↓4.0	↓5.9	↓7.9
	K-Center (SENER; SAVARESE, 2018)	↓0.9	↓1.7	↓4.7	↓4.1	↓6.0	↓8.0
	Least Confidence (COLEMAN et al., 2019)	↓0.6	↓1.1	↓5.3	↓4.0	↓5.9	↓8.4
	Margin (COLEMAN et al., 2019)	↓0.7	↓1.3	↓4.7	↓4.2	↓6.0	↓8.0
	Forgetting (TONEVA et al., 2019)	↓0.9	↓1.5	↓3.9	↓2.9	↓5.1	↓8.3
	GraNd-4 (PAUL et al., 2021)	↓<u>0.3</u>	↓1.0	↓4.4	↓3.6	↓6.8	↓9.4
	DeepFool (DUCOFFE; PRECIOSO, 2018)	↓0.5	↓1.5	↓5.6	↓4.0	↓5.0	↓6.4
	Craig (MIRZASOLEIMAN et al., 2020)	↓0.8	↓3.3	↓7.2	↓3.8	↓6.3	↓8.5
	Glister (KILLAMSETTY et al., 2021)	↓0.4	↓1.6	↓4.7	↓3.6	↓5.0	↓7.8
	EL2N-20 (TONEVA et al., 2019)	↓<u>0.3</u>	↓0.5	↓3.7	↓1.0	↓6.1	-
	DP (YANG et al., 2022)	↓0.7	↓1.8	↓4.8	↓1.0	↓5.1	-
Dynamic	Random (OKANOVIC et al., 2024)	↓0.8	↓1.1	↓2.6	↓0.9	↓2.9	-
	ϵ-greedy (RAJU et al., 2021)	↓0.4	↓0.7	↓<u>1.5</u>	↓1.8	↓3.4	-
	UCB (RAJU et al., 2021)	↓<u>0.3</u>	↓0.9	↓1.7	↓0.9	↓2.9	-
	InfoBatch (QIN et al., 2024)	↑**0.0**	↓<u>0.5</u>	↓**0.9**	↑<u>0.0</u>	↓**0.1**	↓**1.7**
	Random+Ours	↑**0.0**	↓**0.3**	↓1.6	↑**0.2**	↓<u>1.4</u>	↓<u>1.8</u>

The previous experiments confirm that current state-of-the-art methods typically neglect critical learning periods. By successfully addressing this oversight, we fill a significant gap in the literature, demonstrating that respecting these initial phases is essential for maximizing data pruning efficiency without compromising model accuracy.

5.7 Effectiveness on Random Data Pruning

Recent study confirms that random data pruning is surprisingly effective, often matching or outperforming more complex selection criteria when calibrated correctly (OKANOVIC et al., 2024). Building on this insight, we aim to verify the performance of our method when coupling with this data pruning strategy, isolating the specific contribution of our timing strategy when applied to this stochastic baseline, further clarifying how respecting critical periods enhances even the most elementary pruning mechanisms. By applying our timing-aware approach to random selection, we seek to demonstrate that respecting the critical learning period enhances even the simplest data pruning heuristics. For this purpose, we analyze its accuracy on ResNet18 trained on CIFAR-10, applying random data pruning at the standard ratios of 30%, 50%, and 70%.

Table 4 – Model predictive performance comparison on ResNet18/CIFAR-10 with random pruning. Δ_{Acc} (pp) denotes the percentage point difference from the baseline (higher is better).

Method	Pruning Ratio	Δ_{Acc} (pp)
Random	30%	-0.88
Ours + Random	30%	-0.56
Random	50%	-0.30
Ours + Random	50%	-0.24
Random	70%	-2.05
Ours + Random	70%	-0.91

Crucially, we implement a **dynamic random pruning** strategy (OKANOVIC et al., 2024): at each epoch, we select a new random subset of samples, ensuring that the specific data points vary throughout the training process. Our objective is to demonstrate that the method remains effective even when the data selection mechanism is stochastic.

Table 4 presents the Accuracy Delta and time savings achieved when applying our method. The results confirm that our approach improves accuracy while maintaining competitive reductions in computational costs, regardless of the pruning ratio. Specifically, at a 70% pruning rate, we increase accuracy by 1.14 (pp) while incurring only a marginal 5.39% increase in training time compared to standard random pruning, primarily due to the removal of samples from epoch zero. It is interesting to note that even with the marginal increase in training time compared to random pruning (due to the full-data initial phase), this approach remains significantly faster than top-performance methods such as IES (YUAN et al., 2025) and InfoBatch (QIN et al., 2024), offering a superior balance of speed and accuracy.

5.8 The Role of Annealing

Qin et al. 2024 highlight that exposing the model to the full dataset during the final epochs, after applying data pruning, improves accuracy. This exposure allows the model to learn from data points that face exclusion by pruning. To achieve this, they introduce the Annealing parameter $\delta \in (0, 1)$. This parameter controls when the model reverts to the full dataset, that is, we apply pruning only during the first $\delta \cdot N$ epochs, where N is the total number of epochs in training. After that, we train the model on the complete dataset for the remaining epochs. Therefore, higher values of δ lead to a greater reduction in training costs.

In this experiment, we combine the analysis of Qin et al. 2024 with our contributions regarding critical periods to investigate the role of annealing in balancing training cost and

model performance. To integrate both intensive training recipes and data pruning while accounting for critical periods, we first apply data augmentation (with $k = 3$) until the critical period ends at epoch i^* found by our method. Following this, we introduce data pruning with a pruning ratio of 1% ($k = 0.01$) during the $\delta \cdot (N - i^*)$ subsequent epochs.

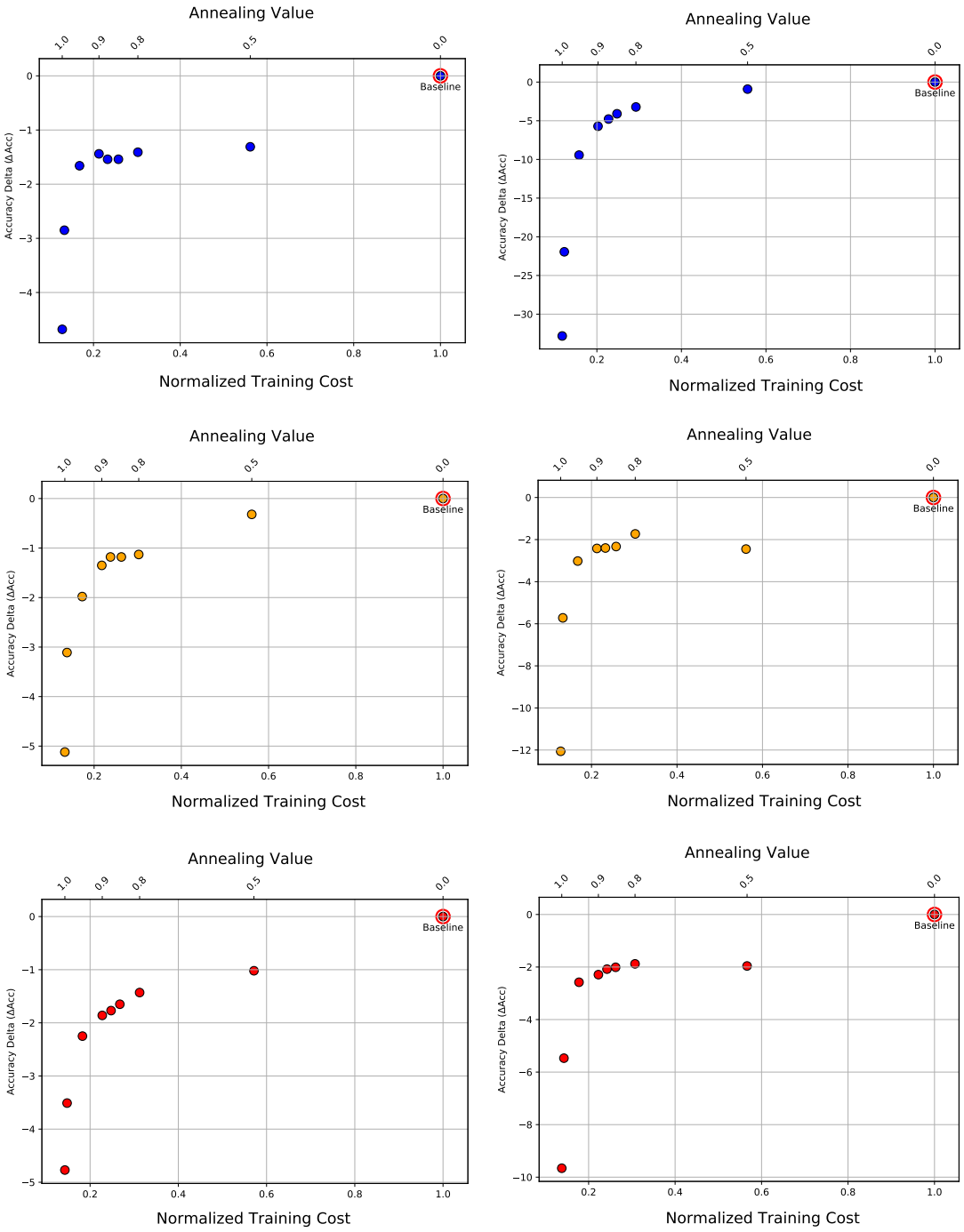


Figure 6 – Impact of annealing on accuracy delta (ΔAcc) in pp and training cost for different ResNet models and datasets. Left column shows samples from CIFAR-10, while right column displays images from CIFAR-100. The horizontal arrangement places ResNet44 in the top row, ResNet56 in the middle, and ResNet110 in the bottom row. The training setup includes data augmentation with $k = 3$ and data pruning with $k = 0.01$.

After this pruning phase, we resume the use of data augmentation until the conclusion of training.

Figure 6 illustrates the impact of varying the annealing parameter, using δ values of 1.0, 0.99, 0.95, 0.9, 0.875, 0.85, 0.8, and 0.5, across different architectures for CIFAR-10/100. The results indicate that reverting to the full dataset for even a few epochs significantly improves accuracy while maintaining low normalized training costs. For example, with a lower annealing value such as 0.5, we reduce training costs by nearly 45%, yet accuracy on the validation set remains close to the baseline. Notably, when $N = 200$ epochs, an annealing value of $\delta = 0.99$ means that only the final epoch employs data augmentation. Compared to the $\delta = 1.0$ (i.e., no annealing) configuration, this subtle change results in an accuracy improvement of up to 2 pp for CIFAR-10 and up to 11 pp for CIFAR-100, while incurring a negligible increase in training cost (less than 1 pp). These findings underscore the role of annealing in reintroducing the full dataset at the end of training to recover lost knowledge with minimal extra cost.

By selecting an appropriate annealing value, one balances normalized training cost and accuracy based on specific requirements, concentrating data points and computational effort during the critical period and at the final refinement stage, enabling the model to better capture and refine knowledge during training.

5.9 Effectiveness on Large Datasets and Optimizers

To validate the effectiveness of our method, we analyze its performance across different datasets and optimizers. Our objective is to demonstrate that the method is agnostic to both dataset and optimizer choices, further reinforcing its general applicability. To this end, we consider the following datasets: CIFAR-10/100 (KRIZHEVSKY et al., 2009a; KRIZHEVSKY et al., 2009b), EuroSAT (HELBER et al., 2018), TinyImageNet (LE; YANG, 2015) and ImageNet30 (HENDRYCKS et al., 2019). On these datasets, the popular architecture is ResNet50; therefore, we employ it in these experiments.

Table 5 introduces the accuracy delta and time saved when applying our method on large datasets. The results confirm that our approach maintains competitive predictive performance while significantly reducing computational costs, independent of the dataset. More concretely, we save up to 65.79% of training time while increasing accuracy on EuroSAT by 1.02 pp.

Similarly, Table 6 examines the impact of different optimizers on the effectiveness of our method in identifying critical periods. This experiment is important to show that our method is neither confined nor biased to the iterative learning process of SGD. Table 6 confirms the consistency of our method across different optimizers, demonstrating its

Table 5 – Accuracy delta (ΔAcc) in percentage points and time saved of our method across different datasets using ResNet50 with augmentation factor $k = 3$. ↑ and ↓ mean accuracy improvement and decrease, respectively.

	CIFAR-10	**CIFAR-100**	**EuroSAT**	**TinyImageNet**	**ImageNet30**
ΔAcc	↓0.32	↓2.36	↑1.02	↓0.20	↑0.34
Saved (%)	49.35	49.47	65.79	60.31	61.70

optimizer-agnostic nature and ensuring applicability in diverse training settings. The results indicate that changing the optimizer has minimal impact on our method. The accuracy deltas remain within a narrow range, and the time saved is consistently around 33% across all optimizers, reinforcing the robustness of our approach.

These results, combined with our previous experiments on different architectures, reinforce that our method is not only dataset-agnostic and optimizer-agnostic but also model-agnostic, further supporting its wide-ranging applicability.

5.10 Computational Benefits and GreenAI

Existing works show that modern models emit high levels of carbon dioxide (CO_2) due to their substantial processing capacity and energy requirements during training and implementation (LACOSTE et al., 2019; FAIZ et al., 2024; MORRISON et al., 2025). However, our approach drastically lowers the carbon footprint through a direct increase in computational efficiency. Specifically, applying our method to ResNet56 results in a 58.89% reduction in CO_2 emissions. Identifying critical periods also reduces financial costs, achieving a 58.33% reduction on this same model. For other architectures, we achieve results nearing a 60% reduction in both CO_2 emissions and financial costs. Within data pruning experiments, applying our method with a pruning ratio of 50% (after the critical period) results in an average reduction of **33.79%** in resource consumption compared to the full dataset. We estimate these values using the Machine Learning Impact Calculator (LACOSTE et al., 2019).

To summarize, our efforts yield significant advancements in GreenAI by effectively lowering carbon footprint and enhancing the financial accessibility of deep learning models.

Table 6 – Accuracy delta (ΔAcc) in percentage points and time saved of our method across different optimizers using ResNet50 on CIFAR-10 with augmentation factor $k = 2$. ↑ and ↓ mean accuracy improvement and decrease, respectively.

	SGD	**AdamW**	**RMSprop**	**AdaGrad**
ΔAcc	↓1.04	↓0.18	↓0.59	↓0.32
Saved (%)	33.75	33.68	32.99	33.57

6 Critical Periods in Transformer-like Architectures

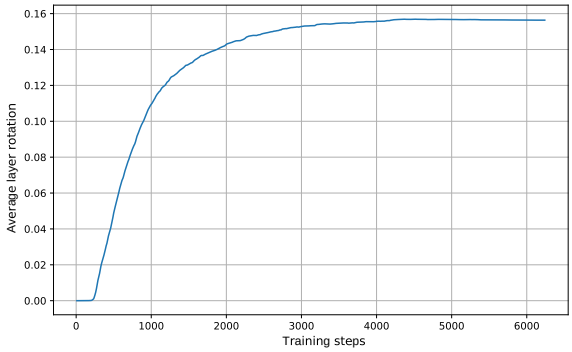
The findings of this research introduce encouraging results, with many promising research opportunities still to address. This chapter outlines the boundaries of our current approach, focusing primarily on the challenges and considerations of extending these methodologies to Large Language Models (LLMs). In this direction, we employ Low-Rank Adaptation (LoRA) as a regularization mechanism within architectures such as BERT (DEVLIN et al., 2019) and RoBERTa (LIU et al., 2019) to mirror the role of data pruning in computer vision. This approach aims to reduce computational overhead via dynamic rank control upon detection of the critical learning period.

Complete fine-tuning of Large Language Models involves high computational costs. Parameter-Efficient Fine-Tuning (PEFT) methods offer a practical alternative: rather than update every weight, these techniques introduce sparse trainable parameters while maintaining a frozen base model (HOULSBY et al., 2019; LI; LIANG, 2021; LESTER et al., 2021). Within this category, Low-Rank Adaptation (LoRA) proposes the decomposition of weight updates ΔW into low-rank factors A and B, where $\Delta W \approx AB^\top$ and rank $r \ll \min(d_{\text{in}}, d_{\text{out}})$ (HU et al., 2022).

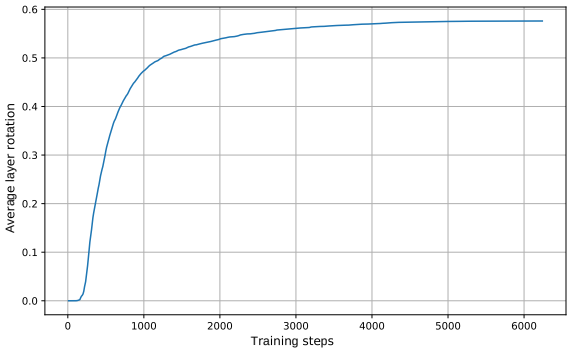
The LoRA architecture minimizes memory consumption and floating-point operations (FLOPs) during optimization, maintains output quality, and enables base model reuse across diverse tasks with minimal cost. Recent extensions integrate LoRA with quantization to further minimize memory and bandwidth requirements, facilitating the deployment of massive models on commodity hardware (DETTMERS et al., 2023; WANG et al., 2025). Notably, QLoRA executes backpropagation through low-rank factors while the base model weights remain in low precision, achieving significant cost reductions with negligible model performance loss (DETTMERS et al., 2023). In practice, the rank r dictates model flexibility: higher values enhance adaptation during early training phases, while lower values serve as regularization and offer computational efficiency in final stages.

However, fundamental differences in training dynamics within the language domain obstruct the efficacy of the current estimator: *Layer Rotation*. Unlike vision tasks that span numerous epochs, LLM fine-tuning occurs over very few cycles—typically two to five (DEVLIN et al., 2019; MOSBACH et al., 2021)—necessitating a shift toward step-level analysis. Figure 7a illustrates that average layer rotation converges to much lower values in this setting, as global weights (i.e. W, A and B) undergo only micro-adjustment during training. While Figure 7b shows that using only LoRA weights (i.e. A and B) yields a more informative signal, convergence remains near 0.5 rather than the unity value from

image classification.



(a) Considering the full weights (W) and LoRA (A and B).



(b) Considering only LoRA (A and B).

Figure 7 – Evolution of the average Layer Rotation throughout training with a *rank* LoRA of 4096 (i.e., $r = 4096$).

Further analysis reveals that layer rotation patterns within transformers remain opposite to those within convolutional networks. Figure 8 shows the evolution of the metric across different layers of the RoBERTa model. Within this small architecture, every layer displays significant values, whereas stability remains more typical for larger models. Additionally, the final classification layer generally maintains lower rotation relative to other components.

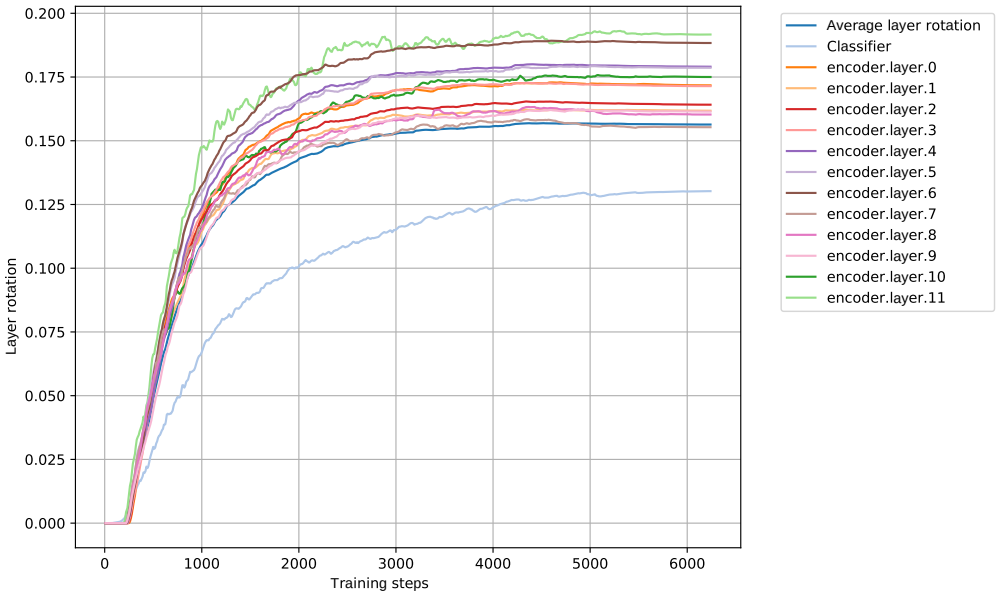


Figure 8 – Layer rotation throughout the steps for the full weight vector with a *rank* of 4096 (i.e., $r = 4096$), where each curve represents a layer.

Furthermore, in the context of critical periods, the initial moments (steps) are the most important. Figure 9 expands the analysis to the initial 78 training steps. Within this interval, every layer—excluding the final classification component—maintains nearly zero rotation, producing a horizontal average curve. This trend opposes the behavior within

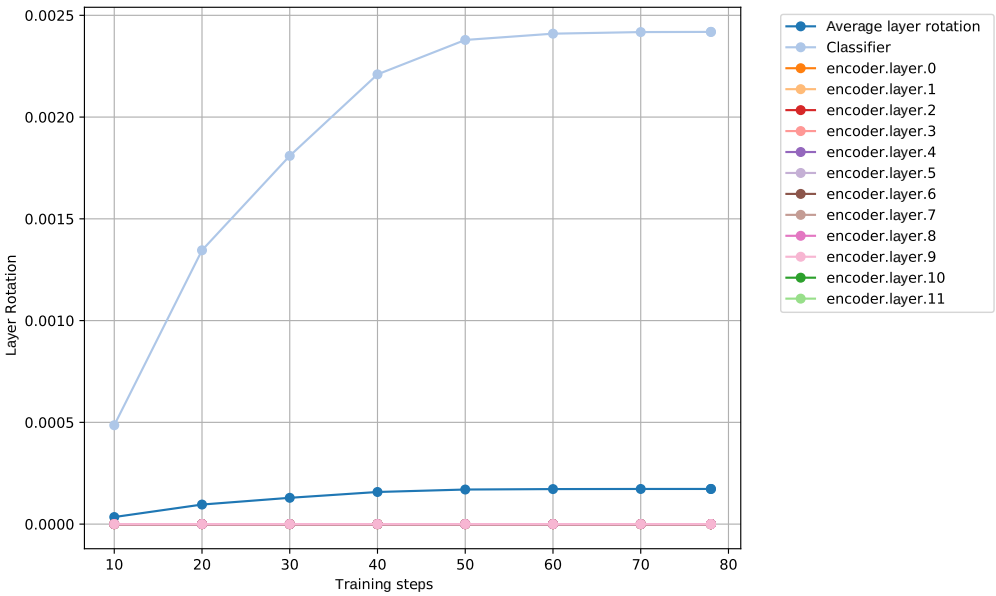


Figure 9 – Layer rotation throughout the training steps of the first training epoch for the full weight vector with a *rank* of 4096 (i.e., $r = 4096$), where each curve represents a layer. It is important to note that most layers maintains nearly zero rotation.

image classification models. Vision models exhibit nearly vertical average rotation curves at the start, reaching a horizontal state only during final epochs near convergence.

In contrast, Figure 10 illustrates that initial layer rotation for LoRA weights displays higher heterogeneity while maintaining a nearly horizontal curve profile. Consequently, this observation necessitates an adaptation of our method for identifying critical periods—presuming a monotonic tangent angle decay within the average rotation curve—for the language model domain and its unique metric dynamics.

Finally, the structural asymmetry of LoRA poses a mathematical challenge. Matrix B begins as a zero vector; this configuration triggers a mathematical void for cosine distance during initialization. While a reference shift to the first epoch weights provides some clarity, Figure 11 demonstrates that matrix A and matrix B follow distinct convergence paths. Overall, the discussion above shows that our method works poorly for LLMs; therefore, there is a need to explore other forms of identifying the critical period for this family of models. Potential improvements include weight-relative rotation measures and detection algorithms with higher resolution to capture the subtle nuances of short, step-intensive transformer training runs.

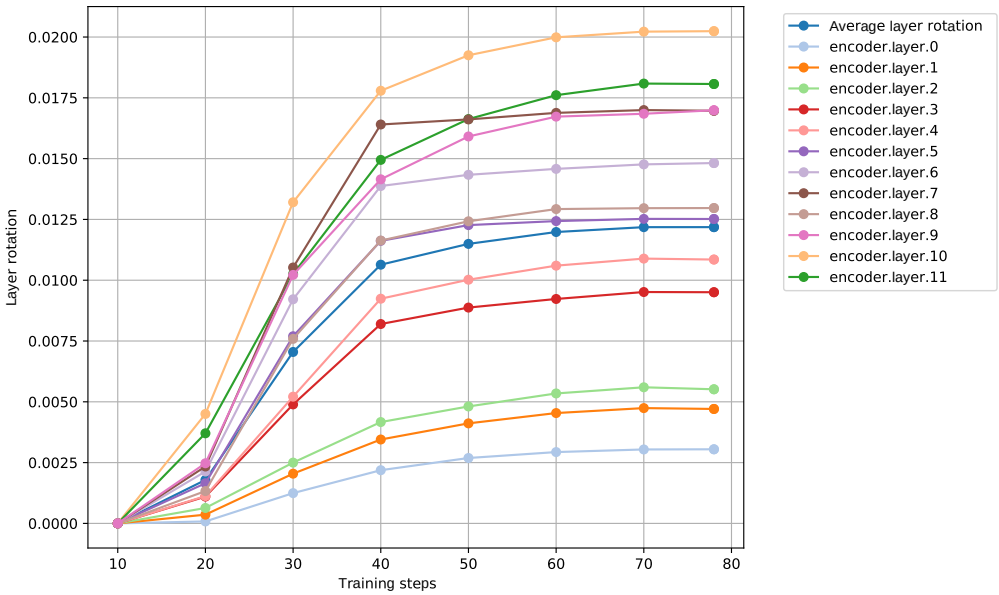
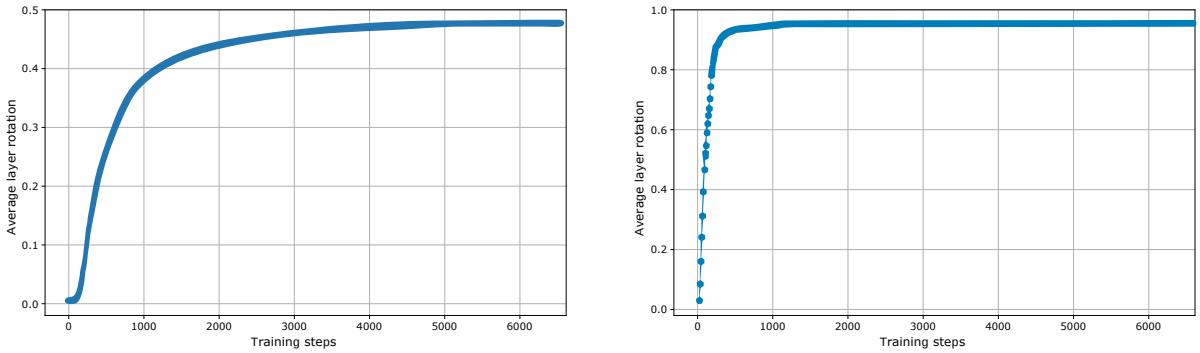


Figure 10 – Layer rotation throughout the training steps of the first training epoch for the LoRA weight vector, with a LoRA rank of 4096 (i.e., $r = 4096$).



(a) Trends in LoRA A layer rotation.

(b) Trends in LoRA B layer rotation.

Figure 11 – Separate contributions from A and B to the average rotation with a LoRA *rank* of 4096 (i.e., $r = 4096$).

7 Conclusions

Existing works suggest that early epochs, known as critical periods, play a decisive role in the success of many training recipes. However, the literature lacks a systematic approach for precisely identifying them. In this work, we propose a systematic method to identify critical learning periods in neural network training. We validate this method within the specific domain of image classification. Our method leverages a quite simple prediction estimation — Layer Rotation — to analyze the generalization behavior during training and, then, identify when critical periods emerge. Importantly, this work answers the question of how to identify critical periods during the course of training.

By formalizing the concept of Critical Learning Periods within the data pruning domain, we demonstrate that the timing of data removal is just as critical as the selection criteria itself. Our method leverages a generalization estimator to pinpoint a potential end of this critical phase. By doing so, it ensures that the model establishes effective foundational features before discarding any data. Importantly, this approach resolves the instability in state-of-the-art methods like IES (YUAN et al., 2025) and InfoBatch (QIN et al., 2024), eliminating the need to re-introduce pruned samples to recover model predictive performance and guaranteeing a stable, monotonic reduction in training complexity. Extensive experiments confirm our research statement: *larger layer rotations* – i.e. as cosine distance between the final and initial weights increases – reliably predict enhanced generalization performance and, hence, indicate the emergence of critical periods.

From a practical perspective, our approach significantly improves training time by applying resource-intensive training recipes (such as data augmentation) only during critical learning periods. As a concrete example, this results in up to 2.5× training cost reductions with minimal accuracy trade-offs across diverse benchmarks. Specifically, the method achieves significant milestones by reducing the training time of popular architectures by up to 59.67%. This leads to a 59.47% decrease in CO_2 emissions and a 60% reduction in financial costs, all without compromising model predictive performance.

Within the data pruning landscape, our strategy sets a new reference for efficient training, achieving near-lossless generalization (i.e., accuracy) across diverse architectures on standard benchmarks. By delaying pruning until the critical period closes, we not only preserve the predictive capacity of the model—limiting accuracy drops to a negligible 0.09%—but also unlock significant computational savings, reducing resource consumption by over 33%. These results directly contribute towards Green AI, offering a practical, scalable solution that lowers the carbon footprint of deep learning without compromising the quality of the final model. Finally, we suggest that the suitable moment to start data

pruning without compromising learning effectiveness is at the end of the critical period. Our findings underscore the untapped potential of early-phase analysis for refining training recipes, offering practical insights for sustainable machine learning and resource allocation. Importantly, our method fills the gap in existing studies on critical learning periods that fail to offer ways to identify them. We hope this work inspires further exploration into adaptive training methods and efficient deep learning practices.

7.1 Future Works

The findings of this research introduce encouraging results, with many promising research opportunities still to address. This section discusses some possible future directions. The first involves the expansion of these methods into the Large Language Model landscape. Within this domain, our method does not work properly as the generalization estimator lacks the sensitivity for subtle training variations. Complex dynamics hide signals and block critical period detection during training. Future investigations involve the exploration of alternative metrics with higher resolution for such architectures. Superior measures potentially reveal these nuances, allowing the application of critical learning period principles to LLM optimization.

Bibliography

ACHILLE, A.; ROVERE, M.; SOATTO, S. Critical learning periods in deep networks. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2019. Cited 5 times in pages 13, 14, 19, 36, and 37.

AGARWAL, S. et al. Contextual diversity for active learning. In: SPRINGER. *European Conference on Computer Vision*. [S.l.], 2020. p. 137–153. Cited in page 41.

BALLESTER, R. et al. Predicting the generalization gap in neural networks using topological data analysis. *Neurocomputing*, 2024. Cited 2 times in pages 13 and 24.

BENGIO, Y. et al. International AI safety report. 2025. Cited 2 times in pages 12 and 14.

BOMMASANI, R. et al. On the opportunities and risks of foundation models. *ArXiv*, 2021. Cited in page 12.

CARBONNELLE, S.; VLEESCHOUWER, C. D. Layer rotation: a surprisingly simple indicator of generalization in deep networks? In: *International Conference on Machine Learning (ICML)*. [S.l.: s.n.], 2019. Cited 3 times in pages 25, 31, and 38.

CAZENAVETTE, G. et al. Dataset distillation by matching training trajectories. *Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited in page 12.

CHEN, Y.; YUILLE, A.; ZHOU, Z. Which layer is learning faster? a systematic exploration of layer-wise convergence rate for deep neural networks. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2023. Cited 4 times in pages 25, 30, 37, and 63.

CHOI, H.; KI, N.; CHUNG, H. W. BWS: best window selection based on sample scores for data pruning across broad ranges. In: *International Conference on Machine Learning (ICML)*. [S.l.: s.n.], 2024. Cited 3 times in pages 22, 28, and 38.

COLEMAN, C. et al. Selection via proxy: Efficient data selection for deep learning. *International Conference on Learning Representations (ICLR)*, 2019. Cited in page 41.

CUBUK, E. D. et al. Randaugment: Practical automated data augmentation with a reduced search space. In: *Neural Information Processing Systems (NeurIPS)*. [S.l.: s.n.], 2020. Cited 3 times in pages 19, 22, and 27.

DENG, W. et al. Non-convex learning via replica exchange stochastic gradient MCMC. In: *International Conference on Machine Learning (ICML)*. [S.l.: s.n.], 2020. Cited in page 36.

DETTMERS, T. et al. Qlora: Efficient finetuning of quantized llms. In: *Neural Information Processing Systems (NeurIPS)*. [S.l.: s.n.], 2023. Cited in page 46.

DEVLIN, J. et al. BERT: pre-training of deep bidirectional transformers for language understanding. In: *Nations of the Americas Chapter of the Association for Computational Linguistics (NAACL-HLT)*. [S.l.: s.n.], 2019. Cited in page 46.

DEVRIES, T.; TAYLOR, G. W. Improved regularization of convolutional neural networks with cutout. *ArXiv*, 2017. Cited 4 times in pages 20, 21, 27, and 28.

DIETTERICH, T. G. Ensemble methods in machine learning. In: *International workshop on multiple classifier systems*. [S.l.: s.n.], 2000. Cited in page 20.

DUCOFFE, M.; PRECIOSO, F. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841*, 2018. Cited in page 41.

ENGSTROM, L.; FELDMANN, A.; MADRY, A. Dsdm: Model-aware dataset selection with datamodels. In: *International Conference on Machine Learning (ICML)*. [S.l.: s.n.], 2024. Cited in page 22.

FAIZ, A. et al. Llmcarbon: Modeling the end-to-end carbon footprint of large language models. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2024. Cited 2 times in pages 12 and 45.

FUKASE, V. Y. et al. One period to rule them all: Identifying critical learning periods in deep networks. *Procedia Computer Science*, v. 264, p. 270–279, 2025. ISSN 1877-0509. International Neural Network Society Workshop on Deep Learning Innovations and Applications 2025. Cited in page 17.

FUKASE, V. Y. et al. Towards efficient training through critical periods. In: *Conference on Graphics, Patterns and Images (SIBGRAPI)*. [S.l.: s.n.], 2025. Cited in page 17.

FUKASE, V. Y. et al. When to prune? the importance of timing in data efficiency training. In: *International Conference on Pattern Recognition (ICPR)*. [S.l.: s.n.], 2026. Cited in page 17.

GOLATKAR, A.; ACHILLE, A.; SOATTO, S. Time matters in regularizing deep networks: Weight decay and data augmentation affect early learning dynamics, matter little near convergence. In: *Neural Information Processing Systems (NeurIPS)*. [S.l.: s.n.], 2019. Cited 6 times in pages 14, 19, 26, 34, 37, and 39.

GRATTAFIORI, A. et al. The llama 3 herd of models. *ArXiv*, 2024. Cited 2 times in pages 12 and 20.

HAN, X. et al. Understanding in-context learning via supportive pretraining data. In: *Association for Computational Linguistics (ACL)*. [S.l.: s.n.], 2023. Cited in page 22.

HE, K. et al. Deep residual learning for image recognition. In: *Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016. Cited 3 times in pages 21, 34, and 36.

HELBER, P. et al. Introducing eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. In: *International Geoscience and Remote Sensing Symposium (IGARSS)*. [S.l.: s.n.], 2018. Cited 3 times in pages 17, 34, and 44.

HENDRYCKS, D. et al. Using self-supervised learning can improve model robustness and uncertainty. In: *Neural Information Processing Systems (NeurIPS)*. [S.l.: s.n.], 2019. Cited 3 times in pages 17, 34, and 44.

HENDRYCKS, D. et al. Pixmix: Dreamlike pictures comprehensively improve safety measures. In: *Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2022. Cited 5 times in pages 19, 20, 21, 22, and 27.

HOULSBY, N. et al. Parameter-efficient transfer learning for NLP. In: *International Conference on Machine Learning (ICML)*. [S.l.: s.n.], 2019. Cited in page 46.

HU, E. J. et al. Lora: Low-rank adaptation of large language models. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2022. Cited in page 46.

KILLAMSETTY, K. et al. Glister: Generalization based data subset selection for efficient and robust learning. In: *Proceedings of the AAAI conference on artificial intelligence*. [S.l.: s.n.], 2021. v. 35, n. 9, p. 8110–8118. Cited in page 41.

KIM, I. et al. Controllable blur data augmentation using 3d-aware motion estimation. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2025. Cited 2 times in pages 19 and 22.

KLEINMAN, M.; ACHILLE, A.; SOATTO, S. Critical learning periods for multisensory integration in deep networks. In: *Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2023. Cited 5 times in pages 13, 14, 19, 34, and 36.

KLEINMAN, M.; ACHILLE, A.; SOATTO, S. Critical learning periods emerge even in deep linear networks. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2024. Cited 5 times in pages 13, 14, 19, 26, and 36.

KRIZHEVSKY, A.; NAIR, V.; HINTON, G. Cifar-10 (canadian institute for advanced research). 2009. Cited 3 times in pages 17, 34, and 44.

KRIZHEVSKY, A.; NAIR, V.; HINTON, G. Cifar-100 (canadian institute for advanced research). 2009. Cited 3 times in pages 17, 34, and 44.

LACOSTE, A. et al. Quantifying the carbon emissions of machine learning. In: *Neural Information Processing Systems (NeurIPS)*. [S.l.: s.n.], 2019. Cited 2 times in pages 12 and 45.

LE, Y.; YANG, X. S. Tiny imagenet visual recognition challenge. In: . [S.l.: s.n.], 2015. Cited 3 times in pages 17, 34, and 44.

LESTER, B.; AL-RFOU, R.; CONSTANT, N. The power of scale for parameter-efficient prompt tuning. In: *Empirical Methods in Natural Language Processing (EMNLP)*. [S.l.: s.n.], 2021. Cited in page 46.

LI, X. L.; LIANG, P. Prefix-tuning: Optimizing continuous prompts for generation. In: *Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL/IJCNLP)*. [S.l.: s.n.], 2021. Cited in page 46.

LI, Z. et al. IDIV: Intrinsic decomposition for arbitrary number of input views and illuminations. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2025. Cited in page 22.

LIU, Y. et al. Roberta: A robustly optimized BERT pretraining approach. *Computing Research Repository (CoRR)*, 2019. Cited in page 46.

LOSHCHILOV, I.; HUTTER, F. Decoupled weight decay regularization. *International Conference on Learning Representations (ICLR)*, 2017. Cited in page 20.

MAHABADI, S.; TRAJANOVSKI, S. Core-sets for fair and diverse data summarization. In: *Neural Information Processing Systems (NeurIPS)*. [S.l.: s.n.], 2023. Cited in page 22.

MAINI, P. et al. Can neural network memorization be localized? In: *International Conference on Machine Learning (ICML)*. [S.l.: s.n.], 2023. Cited in page 19.

MASON-WILLIAMS, G.; DAHLQVIST, F. What makes a good prune? maximal unstructured pruning for maximal cosine similarity. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2024. Cited in page 31.

MAURÍCIO, J.; DOMINGUES, I.; BERNARDINO, J. Comparing vision transformers and convolutional neural networks for image classification: A literature review. *Applied Sciences*, 2023. Cited in page 12.

MIKOŁAJCZYK, A.; GROCHOWSKI, M. Data augmentation for improving deep learning in image classification problem. In: *2018 international interdisciplinary PhD workshop (IIPhDW)*. [S.l.: s.n.], 2018. Cited in page 20.

MIRZASOLEIMAN, B.; BILMES, J.; LESKOVEC, J. Coresets for data-efficient training of machine learning models. In: PMLR. *International Conference on Machine Learning (ICML)*. [S.l.], 2020. p. 6950–6960. Cited in page 41.

MORRISON, J. et al. Holistically evaluating the environmental impact of creating language models. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2025. Cited 2 times in pages 12 and 45.

MOSBACH, M.; ANDRIUSHCHENKO, M.; KLAKOW, D. On the stability of fine-tuning BERT: misconceptions, explanations, and strong baselines. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2021. Cited in page 46.

OKANOVIC, P. et al. Repeated random sampling for minimizing the time-to-accuracy of learning. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2024. Cited 5 times in pages 22, 28, 39, 41, and 42.

PAUL, M.; GANGULI, S.; DZIUGAITE, G. K. Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems (NeurIPS)*, v. 34, p. 20596–20607, 2021. Cited in page 41.

PONS, I. et al. Enhancing distilled datasets via natural data mixing. In: *Conference on Graphics, Patterns and Images (SIBGRAPI)*. [S.l.: s.n.], 2025. Cited in page 12.

QIN, Z. et al. Infobatch: Lossless training speed up by unbiased dynamic data pruning. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2024. Cited 12 times in pages 12, 13, 23, 28, 29, 34, 38, 39, 40, 41, 42, and 50.

RAJU, R. S.; DARUWALLA, K.; LIPASTI, M. Accelerating deep learning with dynamic data pruning. *arXiv preprint arXiv:2111.12621*, 2021. Cited in page 41.

ROBINE, J.; HÖFTMANN, M.; HARMELING, S. Simple, good, fast: Self-supervised world models free of baggage. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2025. Cited in page 19.

SAJADIEH, S. et al. The ai index 2026 annual report. 2026. Cited in page 12.

SANKARARAMAN, K. A. et al. The impact of neural network overparameterization on gradient confusion and stochastic gradient descent. In: *International Conference on Machine Learning (ICML)*. [S.l.: s.n.], 2020. Cited 6 times in pages 24, 30, 37, 59, 60, and 62.

SCHWARTZ, R. et al. Green ai. *Association for Computing Machinery (ACM)*, 2020. Cited in page 12.

SENER, O.; SAVARESE, S. Active learning for convolutional neural networks: A core-set approach. *International Conference on Learning Representations (ICLR)*, 2018. Cited in page 41.

TIAN, Y.; ZHANG, Y. A comprehensive survey on regularization strategies in machine learning. *Information Fusion*, 2022. Cited in page 20.

TONEVA, M. et al. An empirical study of example forgetting during deep neural network learning. 2019. Cited 2 times in pages 23 and 41.

WAGER, S.; WANG, S.; LIANG, P. Dropout training as adaptive regularization. *Neural Information Processing Systems (NeurIPS)*, 2013. Cited in page 20.

WANG, Z. et al. Lora-pro: Are low-rank adapters properly optimized? In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2025. Cited in page 46.

XIA, M. et al. LESS: selecting influential data for targeted instruction tuning. In: *International Conference on Machine Learning (ICML)*. [S.l.: s.n.], 2024. Cited in page 22.

YANG, S. et al. Dataset pruning: Reducing training data by examining generalization influence. *International Conference on Learning Representations (ICLR)*, 2022. Cited in page 41.

YAO, Y.; ROSASCO, L.; CAPONNETTO, A. On early stopping in gradient descent learning. *Constructive approximation*, 2007. Cited in page 20.

YU, R.; LIU, S.; WANG, X. Dataset distillation: A comprehensive review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. Cited in page 12.

YUAN, S. et al. Instance-dependent early stopping. *International Conference on Learning Representations (ICLR)*, 2025. Cited 7 times in pages 12, 13, 29, 38, 39, 42, and 50.

YUN, S. et al. Cutmix: Regularization strategy to train strong classifiers with localizable features. In: *International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2019. Cited 3 times in pages 20, 21, and 27.

ZHANG, H. et al. mixup: Beyond empirical risk minimization. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2018. Cited 4 times in pages 20, 21, 27, and 35.

ZHONG, Z. et al. Random erasing data augmentation. In: *Association for the Advancement of Artificial Intelligence (AAAI)*. [S.l.: s.n.], 2020. Cited in page 35.

A Appendix

A.1 Gradient Confusion

Definition. Gradient confusion quantifies the alignment between the gradients of two distinct batches of data during a specific training epoch. This metric represents the cosine similarity between gradients by the following mathematical expression:

$$CG(A, B) = \frac{A \times B}{\|A\|\|B\|}, CG(A, B) \in [-1, 1],$$

where A and B represent the gradients of two batches of data. A value close to 1 indicates high alignment (low confusion), while a value near -1 reflects high confusion between the gradients.

Experimental setup.

- We use ResNet86 on CIFAR10 standard train/test split (50k/10k).
- We divide the original training samples into batches of $batch\_size$, without transformations.
- For every i in 0, 1, ..., n:
  - We load weights from i epochs of training via standard data augmentation.
  - For each batch, we compute the gradient of the loss with respect to the trainable weights of the model, considering the loaded weights.
  - We compute maximum, minimum, average and median of the pairwise cosine distance and similarity among batches.

Experimental variables. Our study uses these experimental configurations:

- Number of Epochs (n): 200
- Dataset: CIFAR-10
- Model: ResNet86
- Loss Function: Categorical Crossentropy
- Evaluation Metric: Accuracy
- Batch Sizes: {64, 128, 256, 512}

Results. Figure 12 shows the overview of the model accuracies. In this experiment, we permanently reduce data augmentation from $K = 3$ to $K = 1$ after i epochs.

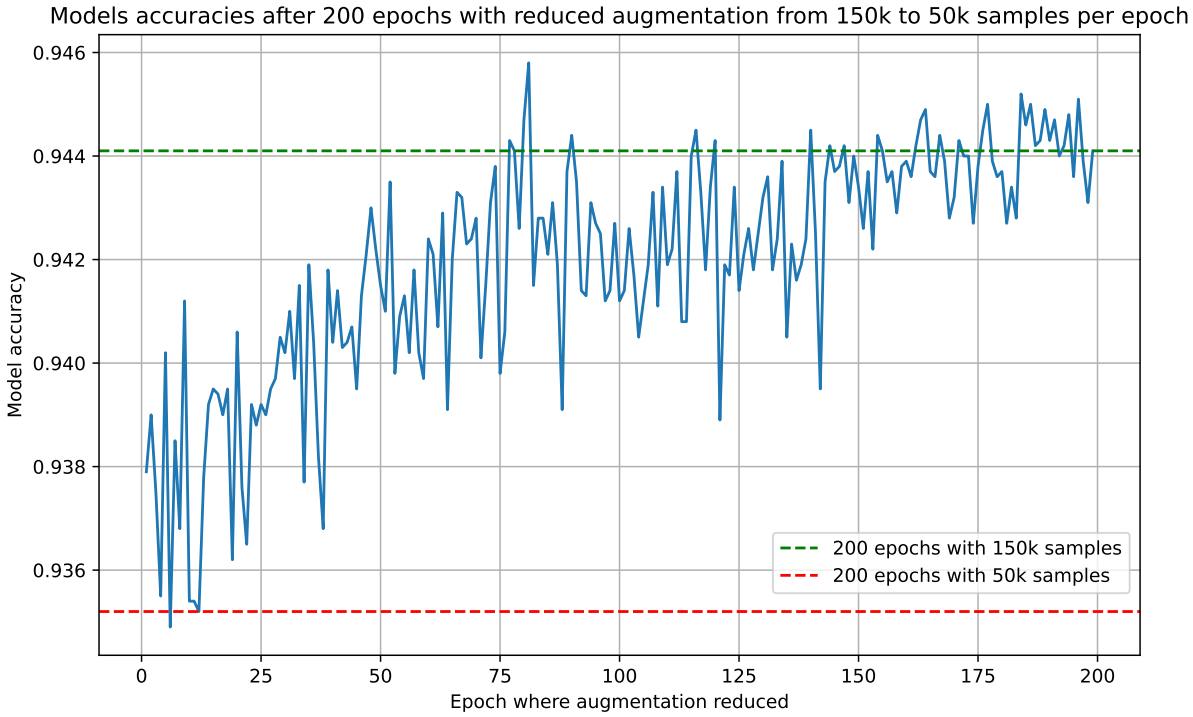
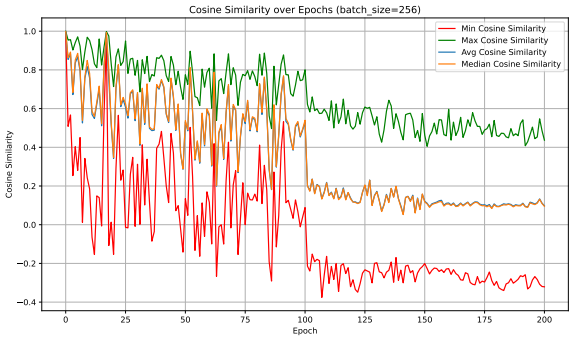
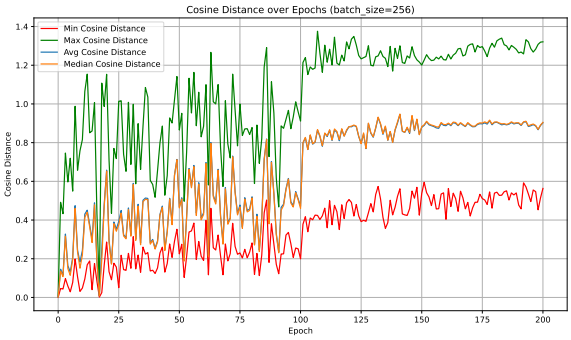


Figure 12 – Overview of the model accuracies.

Figure 13 shows the pairwise cosine distance and similarity among batches for a batch size of 256. The expression $1 - \textit{cosine distance}$ defines *cosine similarity*, ensuring a 100% correlation between both measures. Appendix A.1.1 includes results using other batch sizes.



(a) For cosine similarity, -1, 0 and 1 represents opposite, orthogonal and parallel vectors respectively.



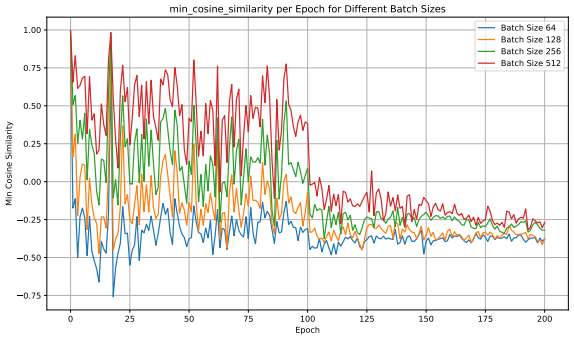
(b) For cosine distance, 2, 1 and 0 represents opposite, orthogonal and parallel vectors respectively.

Figure 13 – Pairwise cosine distance and similarity among batches.

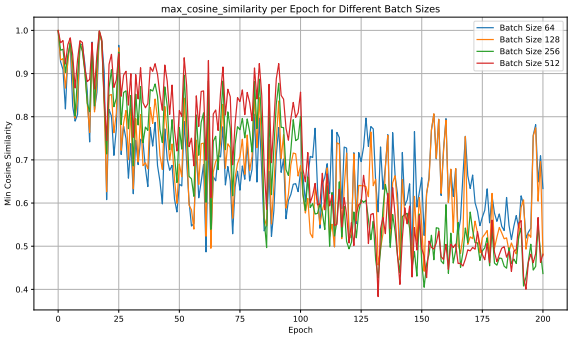
We assume that median and average values analysis are equivalent, as their lines almost completely overlap (see Appendix A.1.2). Therefore, we will stop analyzing the

median. Also, following the work of Sankararaman et. al., we will use only cosine similarity in our analysis from now on (SANKARARAMAN et al., 2020).

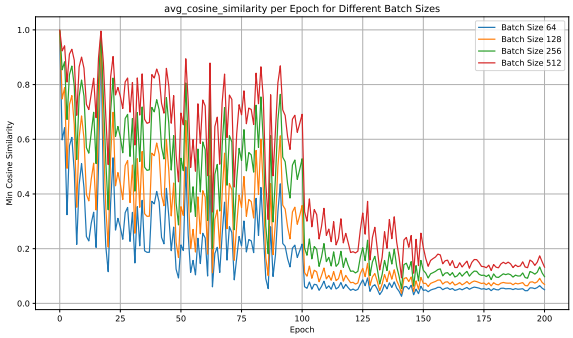
Figure 14 shows the cosine similarity metrics among batches for different batch sizes.



(a) Minimum cosine similarity among batches for different batch sizes.



(b) Maximum cosine similarity among batches for different batch sizes.



(c) Average cosine similarity among batches for different batch sizes.

Figure 14 – Cosine similarity metrics among batches for different batch sizes.

It is interesting to note that for minimum and average cosine similarity plots, the metric for higher batch sizes is always above lower batch sizes. However, this is only true for half of the epochs for maximum cosine similarity, as there is a clear inversion of the trend from epoch 100 onwards. This is also one of the two training points where learning rate is lower. More study of this result is necessary.

Our experiment also indicates that cosine similarity metrics are more volatile for small batch sizes. Sankararaman et. al. (SANKARARAMAN et al., 2020) also observe this when reporting that the variance of the gradient inner product scales down as $\frac{1}{B^2}$, where B is the batch size.

Figure 15 shows the Pearson correlation heatmap between the accuracies of the models from the experiment from Figure 12, where we reduce augmentation on epoch i, and the cosine similarity metrics from the same epoch i.

The correlations are negative and we expect this because a high cosine similarity

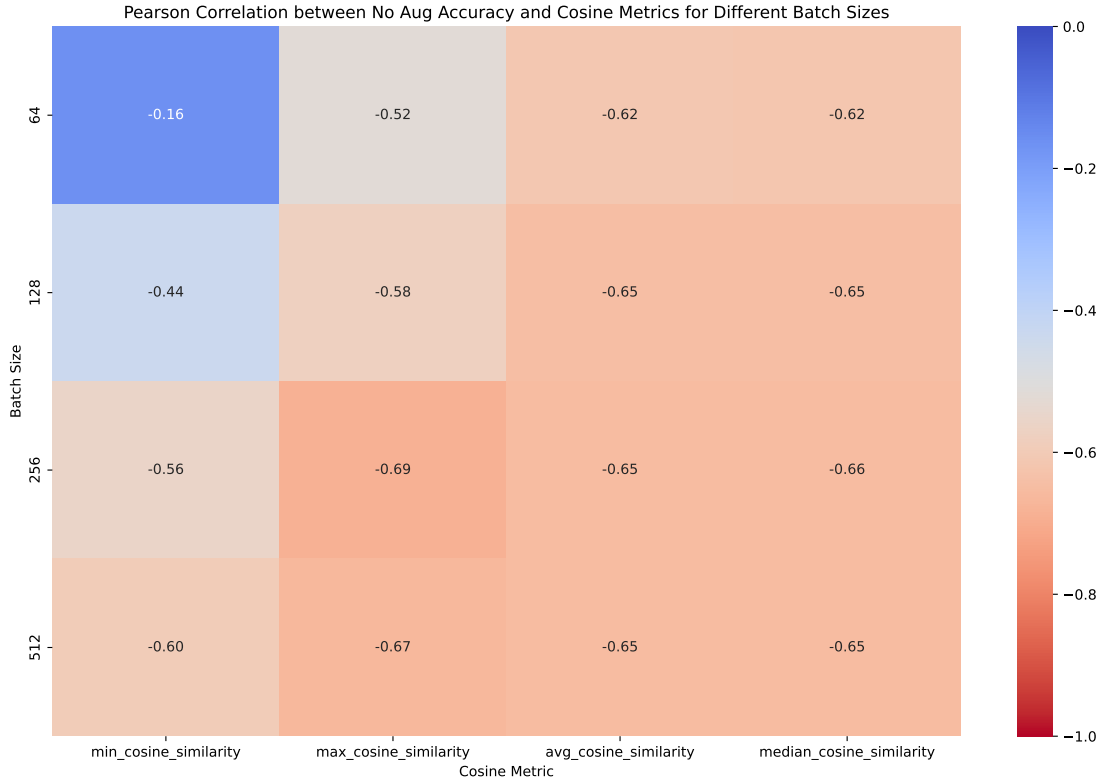


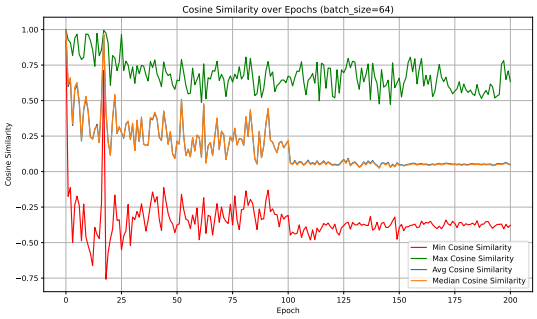
Figure 15 – Pearson correlation heatmap between no augmentation accuracy and cosine similarity metrics.

should mean the model is learning better, so reducing the augmentation during this phase should lead to a lower final accuracy.

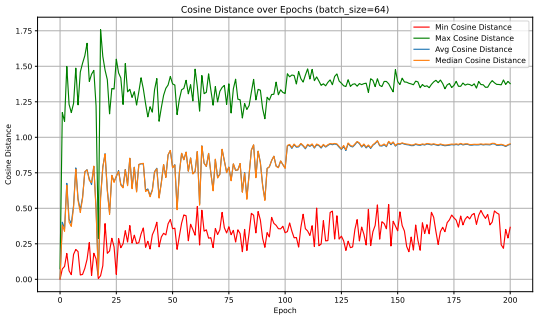
On the other hand, we expect that the higher magnitude correlation came from minimum cosine similarity, as it is the metric that defines Gradient Confusion (SANKARARAMAN et al., 2020). A low minimum cosine similarity indicates that gradients point in diverging directions, so batches disagree on the weight optimization path. However, the maximum and average cosine similarities are the metrics that have the highest correlation with the final accuracy. This is an unexpected result and more study is necessary.

From Figure 15, we could assume that the best metric to identify critical periods is the maximum or average cosine similarities. The average is more consistent across different batch sizes, but the maximum reaches higher magnitude correlations for higher batch sizes, where all similarity metrics are highly correlated (see Appendix A.1.2).

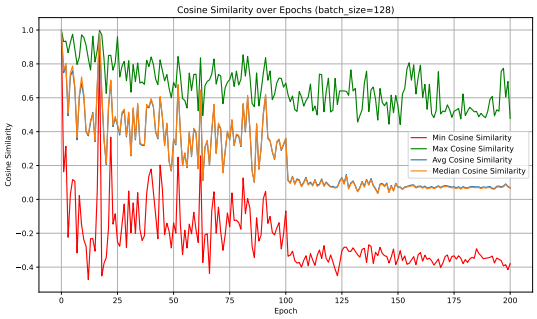
A.1.1 Cosine distance and similarity for batch sizes of 64, 128, and 512



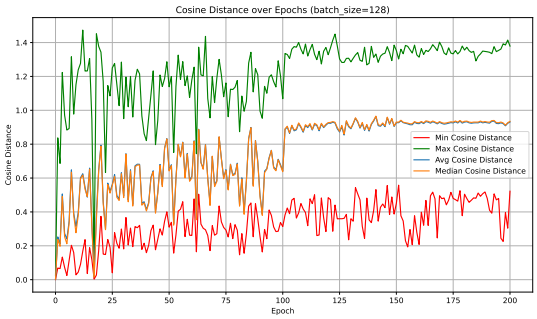
(a) Cosine similarity for batch size 64



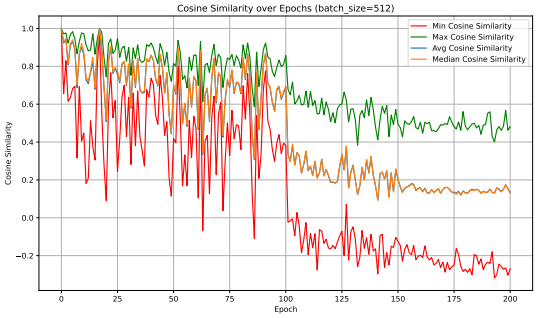
(b) Cosine distance for batch size 64



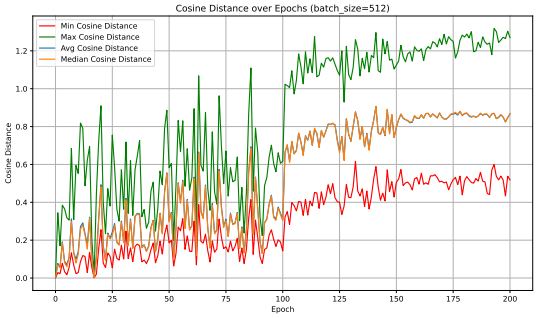
(c) Cosine similarity for batch size 128



(d) Cosine distance for batch size 128



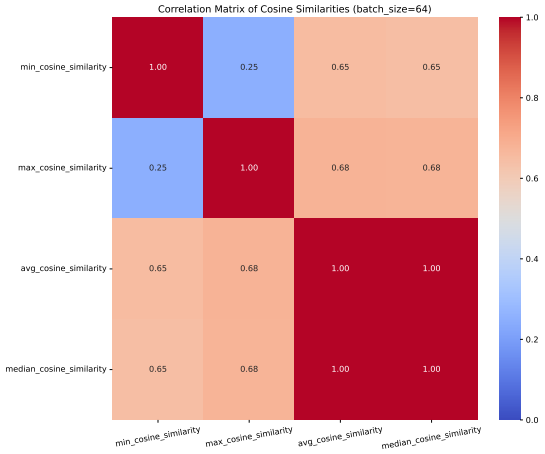
(e) Cosine similarity for batch size 512



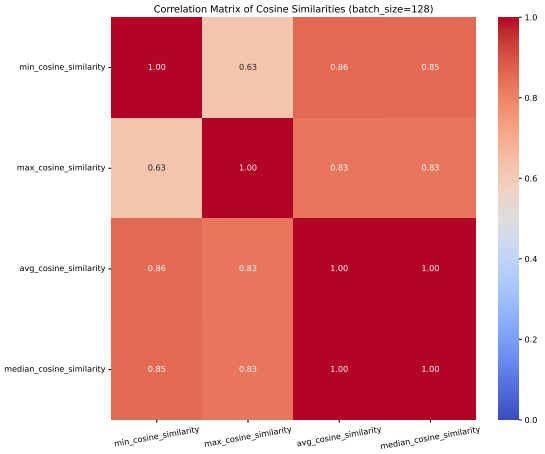
(f) Cosine distance for batch size 512

A.1.2 Correlation between cosine similarity metrics for batch sizes of 64, 128, 256 and 512.

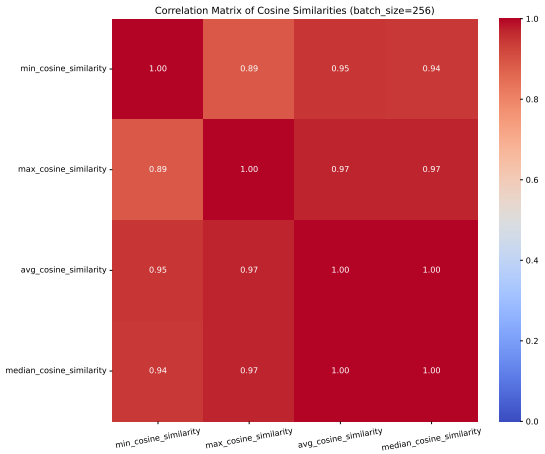
Note that the median and average cosine metrics are almost completely correlated. Therefore, the existing outliers are not enough to dislocate the average from the median. On the other hand, the maximum and minimum cosine metrics are not highly correlated at small batch sizes, but this correlation increases as the batch size increases, to the point of being almost completely correlated at batch size of 512. This may seem unexpected at first, but small batch sizes will raise the variance of the cosine metrics (SANKARARAMAN et al., 2020), so the maximum and minimum values are more likely to be outliers. As the batch size increases, the variance decreases, and the maximum and minimum values are more likely to be closer to the average and median values.



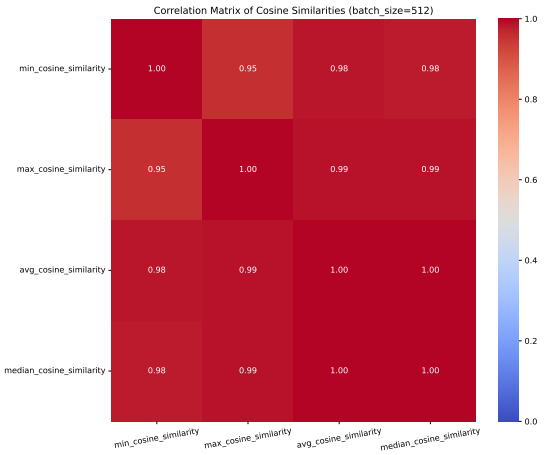
(a) Correlation for batch size 64



(b) Correlation for batch size 128



(c) Correlation for batch size 256



(d) Correlation for batch size 512

Figure 16 – Correlation between cosine distance and similarity metrics for different batch sizes.

A.2 Gradient predictiveness

Definition. Gradient predictiveness measures the consistency between gradients across successive training epochs (CHEN et al., 2023). This metric evaluates the capacity of parameter updates in epoch t to predict updates within epoch $t+1$. High gradient predictability values indicate stable training behavior; conversely, low values suggest abrupt changes in gradient direction, signaling instabilities within the optimization process Formally, using a sequence of gradients G_t from the training process, we define gradient predictability as the cosine similarity between gradients across consecutive epochs:

$$GP(G_t, G_{t-1}) = \frac{G_t \times G_{t-1}}{\|G_t\|\|G_{t-1}\|}, \quad GP(G_t, G_{t-1}) \in [-1, 1],$$

where G_t denotes the gradient vector at time t, while $\cos(G_t, G_{t+1})$ quantifies the directional similarity between successive gradients.

Figure 17 illustrates that during early training stages, gradient predictability exhibits high variability, suggesting model exploration of the loss landscape and parameter adjustments in a less stable manner. Upon a learning rate decay (by a factor of 0.1), we observe a sharp increase in gradient predictability before a nearly linear decline. This behavior indicates that although weight updates initially become more predictable following the learning rate decay, the model gradually loses this stability across training epochs.

Furthermore, results demonstrate that larger batch sizes yield higher gradient predictability relative to smaller batches, primarily within minimum and average cosine similarity metrics. However, the maximum cosine similarity reveals a pattern reversal during specific training phases, notably following the second learning rate decay. This phenomenon indicates that batch size variations impact gradient stability across diverse training stages.

Investigating Gradient Predictiveness validity for critical period identification

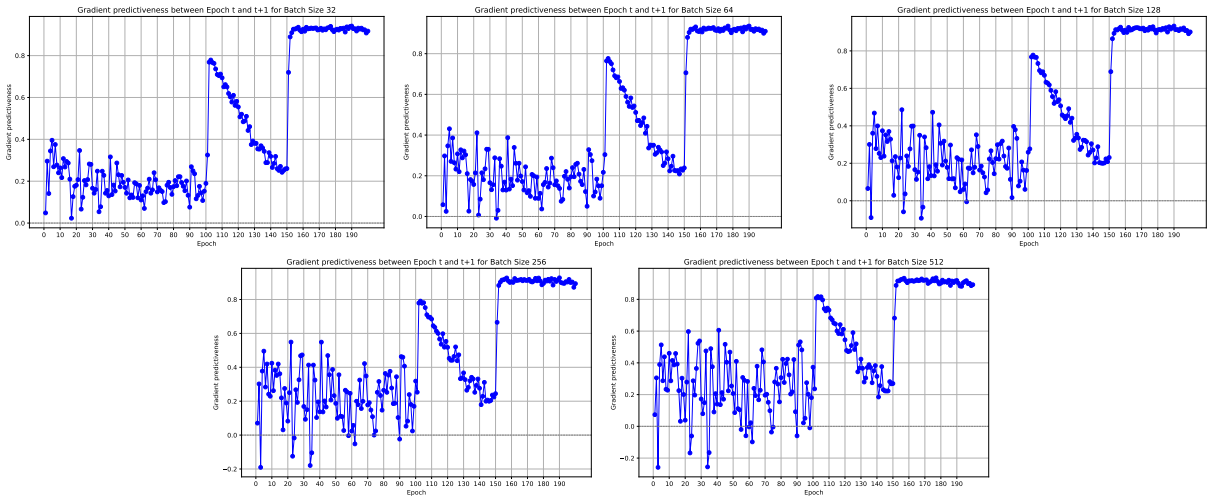


Figure 17 – Gradient Predictiveness over time for different batch sizes.

requires a rigorous analysis of its relationship with final model accuracy. Figure 18 depicts this correlation by displaying gradient predictability and the final accuracy from a complete training cycle where data augmentation stops at time t across diverse batch sizes. This analysis verifies whether a specific Gradient Predictiveness threshold signals the effective moment to terminate strong regularization while maintaining superior model performance.

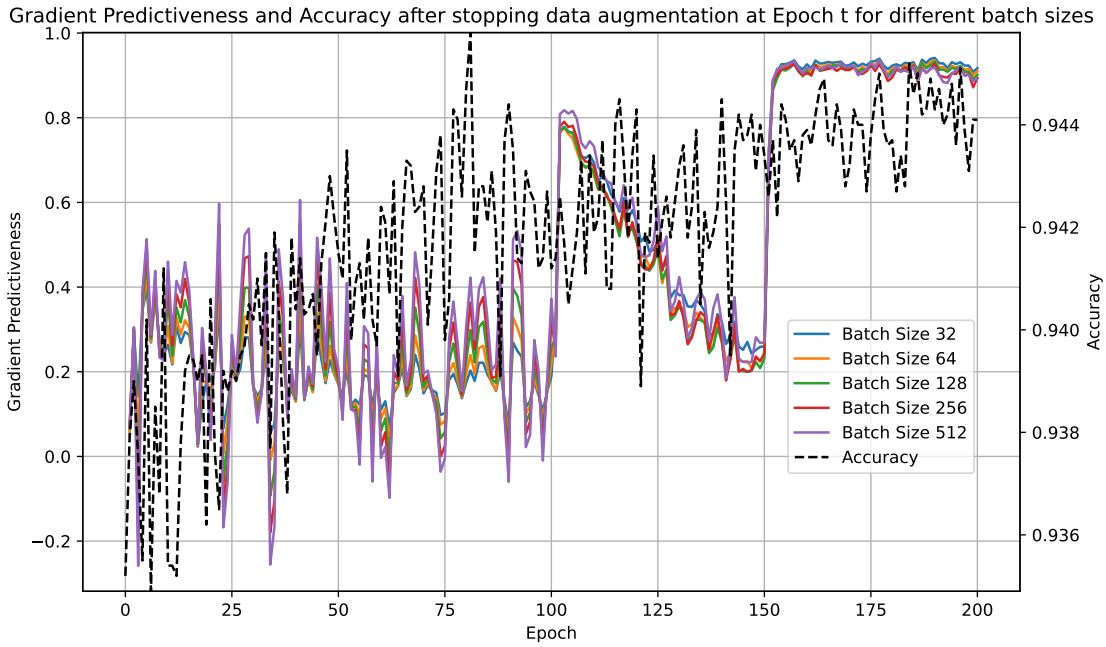


Figure 18 – Gradient predictability and model accuracy when halting data augmentation during training, across epochs for different batch sizes.

Figure 19 illustrates a heatmap of the Pearson correlation between final model accuracy and Gradient Predictiveness across batch sizes, demonstrating a resulting correlation of approximately 0.50 for all configurations. This value indicates a weak relationship between Gradient Predictiveness and final model performance, suggesting that this metric alone fails as a sufficiently reliable predictor of generalization. Consequently, we examine alternative metrics with superior accuracy and stronger links to the generalization capacity of the model.

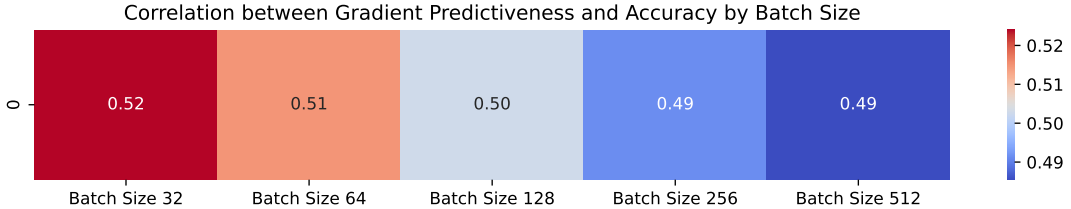


Figure 19 – Pearson correlation between gradient predictability at a given time step and model accuracy after ceasing to use data augmentation at that same time step, by batch size.