Structural Pruning of Convolutional Neural Networks for Efficient and Sustainable Model Optimization

by

Sadegh TOFIGH

MANUSCRIPT-BASED THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
Ph.D.

MONTREAL, "JULY 22, 2026"

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Kim Khoa Nguyen, Thesis supervisor
Department of Electrical Engineering, École de technologie supérieure

Mr. Rami Langar, Chair, Board of Examiners
Department of Software Engineering and IT, École de technologie supérieure

Mr. Ulrich Aïvodji, Member of the Jury
Department of Software Engineering and IT, École de technologie supérieure

Mr. Shervin Vakili, External Examiner
Institut national de la recherche scientifique

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON "JULY 07, 2026"

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

FOREWORD

This dissertation is submitted for the degree of Doctor of Philosophy at the Université du Québec, École de technologie supérieure (ÉTS). The research described herein was conducted under the supervision of Professor Kim-Khoa Nguyen in the Department of Electrical Engineering, during the period from September 2024 to December 2026.

This work is, to the best of my knowledge, original, except where acknowledgments and references are made to previous work. Neither this nor any substantially similar dissertation has been, nor is being, submitted for any other degree, diploma, or qualification at any other university.

Part of this work has been presented in the following publications:

1. S. Tofigh, M. Askarizadeh, M. Omair Ahmad, M. N. S. Swamy and K. K. Nguyen, "Resource-Efficient and Layer Interdependence-Aware CNN Pruning Leveraging Filter Replacement," in IEEE Transactions on Neural Networks and Learning Systems, doi: 10.1109/TNNLS.2026.3658318.
2. S. Tofigh, M. Askarizadeh, M. O. Ahmad, M. N. S. Swamy and K. K. Nguyen, "ACLI: A CNN Pruning Framework Leveraging Adjacent Convolutional Layer Interdependence and γ-Weakly Submodularity," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 48, no. 1, pp. 932-945, Jan. 2026, doi: 10.1109/TPAMI.2025.3610113.
3. S. Tofigh, M. O. Ahmad and M. N. S. Swamy, "A Low-Complexity Modified ThiNet Algorithm for Pruning Convolutional Neural Networks," in IEEE Signal Processing Letters, vol. 29, pp. 1012-1016, 2022, doi: 10.1109/LSP.2022.3164328.

ACKNOWLEDGEMENTS

I wish to express my heartfelt gratitude to everyone who has supported, advised, encouraged, and believed in me throughout this journey. This achievement would not have been possible without the collective wisdom and support of my professors, colleagues, and family.

I would like to extend my deepest thanks to my supervisor, Professor Kim-Khoa Nguyen, for his exceptional research guidance and unwavering support over the years. I am profoundly appreciative of his valuable advice, great understanding, and patience. His mentorship provided me with an outstanding research environment and the opportunities necessary to succeed in my doctoral studies. It has been a true privilege to work under his direction.

I would also like to thank the Chair and the members of the jury for dedicating their time and expertise to the review of this thesis. Their insightful feedback and academic rigor have been invaluable in the final refinement of this work.

Finally, words cannot adequately express my gratitude to my wife, Sara. Thank you for always being there for me, offering your unconditional support, profound understanding, and the invaluable advice that gave me the strength to persevere through the most challenging stages of this research.

élagage structurel des réseaux neuronaux convolutionnels pour une optimisation efficace et durable des modèles

Sadegh TOFIGH

RÉSUMÉ

La prolifération des réseaux neuronaux convolutionnels profonds (CNN) a profondément transformé le paysage de la vision par ordinateur et des systèmes autonomes. Toutefois, les performances de pointe de ces modèles reposent largement sur une sur-paramétrisation extrême, conduisant à un paradigme de « Red AI » caractérisé par une surcharge computationnelle massive et un impact environnemental significatif. Bien que l'élagage structurel se soit imposé comme une solution majeure pour le déploiement de ces modèles sur des dispositifs embarqués aux ressources limitées, les méthodologies actuelles souffrent de trois goulets d'étranglement critiques : une forte dépendance aux données d'entraînement originales pour l'analyse de saillance, un « coût de recherche » élevé induit par des algorithmes gloutons itératifs, et une absence d'ancrage théorique concernant les interdépendances fonctionnelles entre les couches neuronales. Cette thèse répond à ces défis en établissant un cadre de compression structurelle mathématiquement fondé, sans données, et durable.

La contribution théorique centrale de cette recherche consiste à dépasser les heuristiques empiriques fondées sur la magnitude au profit d'une investigation principielle de la propagation du signal. Nous dérivons des bornes supérieures analytiques pour l'erreur absolue moyenne (Average Absolute Error, AAE) propagée d'une couche élaguée vers la suivante. En modélisant le réseau comme une séquence de variétés interdépendantes plutôt que comme un ensemble de couches isolées, ce cadre caractérise l'« impact en cascade » des modifications structurelles. Nous démontrons en outre la propriété de γ-weak de notre fonction d'importance, fournissant une garantie mathématique rigoureuse quant à la stabilité et à la convergence du processus de sélection. Cette base théorique permet d'identifier la redondance en exploitant uniquement les propriétés algébriques intrinsèques des tenseurs de poids, rendant possible un élagage entièrement sans données qui préserve la fidélité représentationnelle du réseau sans nécessiter l'accès à des distributions d'entraînement sensibles.

Afin de résoudre les inefficacités computationnelles de l'élagage traditionnel, cette thèse introduit un paradigme de sélection oblivious (en un seul passage). Les algorithmes gloutons conventionnels, bien que localement optimaux, imposent une taxe de complexité en raison de leurs exigences d'inférence itérative. Nos travaux mettent en évidence un « écart complexité-performance », démontrant qu'un algorithme oblivious bien informé peut atteindre une parité de précision prédictive avec les variantes gloutonnes pour un coût temporel nettement inférieur. Cette efficacité garantit que le processus d'optimisation lui-même demeure durable, évitant que l'empreinte carbone de la phase d'élagage ne dépasse les économies énergétiques réalisées lors de l'inférence du modèle.

Une innovation structurelle majeure présentée dans ce travail réside dans la transition d'un paradigme binaire d'« élagage par élimination » vers une stratégie plus fine de remplacement de

filtres non nuls. Dans de nombreuses régions architecturales à forte sensibilité, la suppression totale d'un filtre (remplacement par un « filtre nul ») entraîne un effondrement catastrophique de la variété du signal. Nous proposons un cadre d'optimisation qui substitue les filtres redondants par des alternatives mathématiquement optimisées et de dimension inférieure. Cette stratégie de remplacement préserve le « flux de connaissance » au sein du réseau et fournit une initialisation architecturale plus stable. Elle s'avère particulièrement efficace dans des environnements contraints en données ou dans des scénarios où le ré-entraînement post-élagage est limité, offrant un compromis supérieur entre parcimonie structurelle et précision.

Au-delà des performances algorithmiques, cette thèse opérationnalise le concept de durabilité environnementale à travers l'introduction de la métrique d'efficacité des ressources (Resource Efficiency, RE). La métrique RE fournit un protocole standardisé pour quantifier le cycle de vie computationnel total de l'optimisation du modèle, en prenant en compte l'analyse de saillance, la sélection et la phase de récupération. Nous proposons également un modèle d'empreinte carbone indépendant du matériel afin d'estimer les équivalents CO_2 (CO_2e) du cycle de vie de l'élagage. Cette approche déplace l'évaluation des algorithmes de compression d'indicateurs statiques de vitesse d'inférence vers une analyse complète du cycle de vie, garantissant que le développement d'une IA efficiente soit lui-même un processus efficient. Les méthodologies proposées ont été validées sur un ensemble de bancs d'essai standards, incluant les architectures VGG, ResNet et MobileNet, évaluées sur les jeux de données CIFAR-10 et ImageNet. Les résultats démontrent que le cadre sans données proposé atteint de manière systématique des performances de pointe (state-of-the-art, SOTA), en maintenant une haute précision même sous des taux d'élagage agressifs. Les résultats confirment que la sélection oblivious et le remplacement non nul constituent une chaîne d'optimisation robuste, rapide et simple.

L'impact de cette recherche est attesté par sa diffusion dans des revues académiques de premier plan. Les dérivations théoriques et les paradigmes sans données ont fait l'objet d'une évaluation par les pairs et ont été publiés dans IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), IEEE Transactions on Neural Networks and Learning Systems (TNNLS) et IEEE Signal Processing Letters. Ces publications établissent le cadre proposé comme une contribution fondamentale au domaine de l'intelligence artificielle durable.

Cette thèse propose ainsi une solution complète aux inefficacités de la compression moderne des réseaux neuronaux. En comblant le fossé entre la théorie de l'optimisation mathématique et l'ingénierie pratique, ce travail établit un nouveau standard pour une optimisation de modèles sans données, consciente des interdépendances et respectueuse de l'environnement. Le cadre développé constitue un outil essentiel pour la prochaine génération de systèmes intelligents haute performance, permettant le déploiement d'architectures neuronales complexes en périphérie tout en respectant les impératifs globaux de la « Green AI ».

Mots-clés: élagage structurel, IA verte, compression sans données, dépendance inter-couches, remplacement de filtres, efficacité des ressources

Structural Pruning of Convolutional Neural Networks for Efficient and Sustainable Model Optimization

Sadegh TOFIGH

ABSTRACT

The proliferation of Deep Convolutional Neural Networks (CNNs) has fundamentally transformed the landscape of computer vision and autonomous systems. However, the state-of-the-art performance of these models is largely predicated on extreme over-parameterization, leading to a "Red AI" paradigm characterized by massive computational overhead and significant environmental impact. While structural pruning has emerged as a primary solution for deploying these models on resource-constrained edge devices, current methodologies suffer from three critical bottlenecks: a heavy reliance on original training data for saliency analysis, an expensive "search tax" incurred by iterative greedy algorithms, and a lack of theoretical grounding in the functional interdependencies between neural layers. This dissertation addresses these challenges by establishing a mathematically grounded, data-free, and sustainable framework for structural model compression.

The core theoretical contribution of this research is the move away from empirical, magnitude-based heuristics toward a principled investigation of signal propagation. We derive analytical upper bounds for the Average Absolute Error (AAE) propagated from a pruned layer to its successor. By modeling the network as a sequence of interdependent manifolds rather than isolated layers, this framework characterizes the "cascading impact" of structural modifications. We further prove the γ-weak property of our importance function, which provides a rigorous mathematical guarantee for the stability and convergence of the selection process. This theoretical foundation allows the framework to identify redundancy using only the intrinsic algebraic properties of weight tensors, facilitating a completely data-free pruning process that preserves the network's representational fidelity without requiring access to sensitive training distributions.

To resolve the computational inefficiencies of traditional pruning, this thesis introduces an oblivious (single-pass) selection paradigm. Conventional greedy algorithms, while locally optimal, impose complexity tax due to their iterative re-inference requirements. Our research identifies a "complexity-performance gap," demonstrating that a well-informed oblivious algorithm can achieve predictive accuracy parity with greedy variants at a fraction of the temporal cost. This efficiency ensures that the optimization process itself is sustainable, preventing the carbon footprint of the pruning phase from outweighing the energy savings gained during model inference.

A significant structural innovation presented in this work is the transition from binary "pruning-as-elimination" to a more granular strategy of non-zero filter replacement. In many high-sensitivity architectural regions, the total removal of a filter (replacement with a "zero-filter") leads to a catastrophic collapse of the signal manifold. We propose an optimization-based framework that substitutes redundant filters with mathematically optimized, lower-dimensional alternatives.

This replacement strategy preserves the "knowledge flow" of the network, providing a more stable architectural initialization. This approach is particularly effective in data-constrained environments or scenarios where post-pruning fine-tuning is restricted, offering a superior trade-off between structural sparsity and accuracy.

Beyond algorithmic performance, this thesis operationalizes the concept of environmental sustainability through the introduction of the Resource Efficiency (RE) metric. The RE metric provides a standardized protocol to quantify the total computational lifecycle of model optimization, accounting for saliency analysis, selection, and recovery. We further propose a hardware-agnostic carbon footprint model to estimate the CO_2 equivalents (CO_2e) of the pruning lifecycle. This shifts the evaluation of compression algorithms from static inference-speed benchmarks to a comprehensive lifecycle analysis, ensuring that the development of efficient AI is itself an efficient process.

The proposed methodologies were validated across a spectrum of standard benchmarks, including VGG, ResNet, and MobileNet architectures, evaluated on the CIFAR-10 and ImageNet datasets. The results demonstrate that the proposed data-free framework consistently achieves state-of-the-art (SOTA) performance, maintaining high accuracy even at aggressive pruning scales. The findings verify that oblivious selection and non-zero replacement provide a robust, fast, and simple pipeline for model optimization.

The impact of this research is evidenced by its dissemination in premier academic venues. The theoretical derivations and data-free paradigms have been peer-reviewed and published in IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), IEEE Transactions on Neural Networks and Learning Systems (TNNLS), and IEEE Signal Processing Letters. These publications establish the framework as a fundamental contribution to the field of sustainable artificial intelligence.

This dissertation provides a comprehensive solution to the inefficiencies of modern neural network compression. By bridging the gap between mathematical optimization theory and practical engineering, the research establishes a new standard for data-blind, interdependency-aware, and environmentally responsible model optimization. The framework serves as a vital tool for the next generation of high-performance intelligence, enabling the deployment of complex neural architectures on the edge while adhering to the global imperatives of "Green AI."

Keywords: structural pruning, green AI, data-free compression, inter-layer dependency, filter replacement, resource efficiency

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

Page

LIST OF ABBREVIATIONS

ETS	École de Technologie Supérieure
CNN	Convolutional Neural Network
AAE	Average Absolute Error
RE	Resource Efficiency
IoT	Internet of Things
FLOPs	Floating-point operations
IB	Information Bottleneck
IB	Information Bottleneck
MI	Mutual Information
HSIC	Hilbert-Schmidt Independence Criterion
SOTA	State-of-the-art

LIST OF SYMBOLS AND UNITS OF MEASUREMENTS

$F^l_{\{i\}F}$	The i-th filter of the l-th convolutional layer.
$F^l[:, :, :, \mathtt{i}]$	The i-th filter of the l-th convolutional layer.
$F^l_{\{j\}C}$	The j-th channel in the l-th convolutional layer.
$F^l[:, :, \mathtt{j}, :]$	The j-th channel in the l-th convolutional layer.
X_l	The output of the l-th convolutional layer.
$\hat{X}^{p_r}_{l+1}$	The output of $(l+1)$-th convolutional layer when the p_r^{th} filter from the l-th convolutional layer is removed.
$\|\cdot\|_2$	ℓ_2-norm.
$\hat{\mathcal{L}}(i)$	Importance of filter i.
H_l	Number of filters in the l-th convolutional layer.
W_l	Width of the output of the l-th convolutional layer.
L_l	Length of the output of the l-th convolutional layer.
r	Pruning ratio.
γ	Weak submodularity parameter.

INTRODUCTION

As the demand for deploying artificial intelligence on "the edge"—including smartphones, IoT devices, and wearables—continues to accelerate, model compression for Convolutional Neural Networks (CNNs) has emerged as a critical research frontier. While traditional CNN architectures like ResNet or VGG have set benchmarks for accuracy in computer vision, their structural complexity often conflicts with the rigorous constraints of real-world hardware. These models typically house millions or even billions of parameters, demanding a memory footprint and storage capacity that far exceed the limits of mobile processors or embedded sensors.

Beyond physical storage, the computational density of large CNNs presents a significant barrier to real-time execution. In safety-critical applications such as autonomous driving or real-time surveillance, inference speed is not merely a performance metric but a fundamental requirement. The billions of floating-point operations (FLOPs) inherent in uncompressed networks introduce latency that can compromise system responsiveness.

Furthermore, the energy profile of massive neural networks poses both practical and ethical challenges. On portable hardware, the intensive CPU and GPU cycles required for inference rapidly deplete battery life, limiting the utility of the device. From a broader perspective, scaling these unoptimized models in data centers results in substantial electricity consumption. Thus, compression serves as a vital tool for environmental sustainability by creating "greener" AI that requires less power per prediction.

Finally, model compression addresses the dual hurdles of connectivity and data integrity. When a model is too cumbersome to run locally, data must be offloaded to the cloud, a process that is often hampered by limited bandwidth—especially when transmitting high-resolution video. By enabling on-device processing, compression ensures that sensitive information, such as private documents or facial data, remains within the user's local hardware, providing a robust layer of privacy and security that cloud-based alternatives cannot guarantee.

Researchers generally categorize these techniques into four primary approaches.

Low-Rank Factorization: Since CNN weights are often stored in high-dimensional tensors, this approach exploits the inherent redundancy within these tensors. It uses mathematical decompositions—such as Singular Value Decomposition (SVD) or Tucker Decomposition—to approximate the original large weight matrix with a product of smaller, lower-rank matrices. This significantly reduces the number of parameters and floating-point operations (FLOPs), though it requires a careful trade-off between the rank (compression rate) and model accuracy.

Transferred/Compact Convolutional Filters: This method focuses on designing or learning specialized filter structures that are naturally efficient. Instead of learning every weight in a standard 3×3 filter, this approach might use spatial transformations or separable convolutions (like those found in MobileNet) to create "compact" versions of the filters. By reusing certain filter patterns across different layers or orientations, the network can maintain its feature-extraction power while using far fewer unique parameters.

Knowledge Distillation: In this "teacher–student" framework, a large, pre-trained "teacher" model is used to train a much smaller "student" model. Rather than just learning from the raw labels (hard targets), the student learns to mimic the teacher's output probabilities (soft targets). These soft targets provide a "knowledge-rich" signal that helps the student model achieve performance levels that it likely could not have reached through standard training alone.

Pruning and Sharing: This approach is often considered the most powerful and intuitive compression strategy, as it directly addresses the over-parameterization of neural networks. Pruning involves identifying and removing unimportant connections or neurons. In unstructured pruning, individual weights with small magnitudes are zeroed out, resulting in sparse matrices. However, since standard hardware often struggles with irregular sparsity, researchers frequently use structured pruning, which removes entire filters, channels, or layers. This provides a direct

boost to inference speed on most GPUs. Weight sharing, in contrast, avoids storing a unique value for every connection by grouping similar weights together. Using techniques like K-means clustering, the model can force multiple connections to share a single weight value stored in a lookup table. During inference, the model only needs to store the cluster indices and a small table of values, which drastically shrinks the storage footprint.

In the technical landscape of CNN optimization, the distinction between unstructured and structured pruning is primarily a trade-off between mathematical precision and hardware-level acceleration. Unstructured, or fine-grained, pruning targets individual weights regardless of their position by removing the least significant connections throughout the network. While this method can achieve extremely high sparsity with minimal accuracy loss, it produces sparse weight matrices that standard hardware like GPUs and CPUs cannot easily accelerate. Because these processors are optimized for dense matrix multiplication, unstructured sparsity often requires specialized software or hardware to achieve real-world speed improvements.

In contrast, structured, or coarse-grained, pruning removes entire structural units such as filters, channels, or layers. By deleting a complete filter, the corresponding feature map is eliminated, which directly reduces the physical dimensions of the tensors. This results in smaller, dense matrices that off-the-shelf hardware can process much faster without specialized support. While structured pruning is more “hardware-friendly” and provides immediate latency benefits, it is a blunter tool that can sometimes lead to a more significant drop in network performance compared to the precision of unstructured methods.

Beyond the final architecture, resource efficiency during the pruning process itself is a major consideration. Traditional iterative pruning requires a repetitive cycle of removing weights and then retraining the model to recover lost accuracy. This cycle can be incredibly time-consuming and energy-intensive, often matching or exceeding the cost of the initial training. To improve sustainability, researchers are developing one-shot pruning methods and saliency criteria based

on mathematical approximations like Taylor expansions, which identify redundant weights without requiring exhaustive empirical testing.

0.1 Research Motivation and Challenges

Designing a pruning method involves a complex optimization problem where the goal is to maximize sparsity without compromising the network's predictive power. Incorporating network performance, specifically the accuracy of the pruned model, introduce several nuanced challenges to the design process.

0.1.1 The Accuracy-Sparsity Trade-off

The fundamental challenge in Convolutional Neural Network (CNN) filter pruning lies in navigating the Pareto frontier between model complexity and predictive performance. As pruning aggressively removes structural components to satisfy stringent hardware constraints, the network's expressive capacity, its ability to represent and extract high-dimensional features, is inevitably diminished. This reduction in capacity often results in a significant degradation of accuracy. Identifying the critical threshold where performance begins to collapse remains an open challenge, as this limit is highly sensitive to both the specific architectural topology (e.g., the residual connections in ResNet, versus the plain structure of VGG) and the complexity of the target task.

In the context of structured pruning, the procedure is fundamentally framed as a constrained optimization problem, where the objective is to minimize the model's size while bounding the loss in performance. However, solving the global optimization problem based directly on the network's final output, as seen in computationally intensive methods, like ThiNet, often leads to prohibitive algorithmic complexity. Consequently, most state-of-the-art pruning frameworks are instead formulated around suboptimal proxy objectives. These methods typically focus

on the derivation of a robust importance function or saliency metric. By accurately ranking the contribution of individual filters, these suboptimal approaches aim to maintain maximum accuracy even under high compression ratios, striking a balance between theoretical optimality and practical implementation.

0.1.2 Time Consumption and the Retraining Cycle

To preserve competitive network performance, filter pruning is seldom executed as a one-shot operation. Instead, state-of-the-art frameworks typically employ an iterative pruning-and-fine-tuning pipeline to mitigate the degradation of the model's representational power. In this regime, only a marginal fraction of filters, determined by a predefined sparsity schedule, is removed during a single iteration. This truncation is immediately followed by a fine-tuning phase, which re-optimizes the remaining parameters to compensate for the localized loss in feature detection.

While this incremental recovery mechanism is essential for maintaining high accuracy at high compression ratios, it introduces significant computational latency. The requirement for repeated training cycles over multiple iterations results in a total optimization time that often exceeds the duration of initial model training. This temporal overhead represents a substantial bottleneck, particularly for on-device pruning, where limited energy budgets and real-time constraints make such exhaustive retraining cycles practically prohibitive.

0.1.3 Data Dependency

A significant limitation inherent in the majority of state-of-the-art (SOTA) pruning frameworks is their fundamental reliance on data-dependent filter selection. This dependency poses a dual challenge for practical deployment, particularly in resource-constrained environments. First, from a hardware perspective, the requirement for the original training dataset, or a substantial

representative subset, imposes a prohibitive memory footprint on edge devices, where storage capacity is often insufficient to house large-scale datasets.

Second, the mechanism of filter evaluation typically necessitates multiple forward passes of the data through the network to generate the activations or gradients required for saliency scoring. This procedural requirement significantly scales the computational complexity and energy consumption of the pruning process. Consequently, the reliance on external data not only hinders autonomous on-device pruning but also complicates the optimization pipeline, creating a need for more efficient, data-agnostic, or "data-free" pruning alternatives that can operate using only the inherent structural statistics of the pre-trained weights.

Empirical evidence suggests that the move toward data-free or data-independent pruning algorithms introduces a secondary optimization conflict: the data-efficiency trade-off. While eliminating the dependency on external datasets resolves memory and privacy constraints, it frequently results in a marked decline in the pruned network's performance. Without access to the underlying data distribution, pruning heuristics must rely solely on the model's weight statistics, which often fail to capture the dynamic importance of filters during actual inference.

Consequently, this creates a significant tension between the practical feasibility of the pruning process and the ultimate utility of the compressed model. High-fidelity pruning typically requires the rich signal provided by the training data to calibrate filter saliency and guide the fine-tuning process. In contrast, data-free methods prioritize operational independence over representational accuracy. This necessitates a strategic trade-off for researchers: eliminating data dependencies during filter selection while maintaining acceptable performance and adhering to the resource constraints of the deployment environment.

0.1.4 Sustainability

The “Green AI” paradox highlights a critical contradiction in the lifecycle of neural network compression: the pursuit of environmental sustainability at the point of deployment often necessitates a massive environmental expenditure during the development phase. While one of the primary objectives of filter pruning is to reduce the carbon footprint of inference, enabling models to run on low-power edge devices with minimal energy consumption, the process of arriving at such an optimized architecture is itself extraordinarily energy-intensive.

This paradox is rooted in two main computational bottlenecks:

The Search Space Explosion: Finding the “perfect” sparse architecture often involves a high-dimensional search. The filter selection process may require thousands of GPU hours, and the electricity consumed by high-end data centers used to power these GPU clusters can offset the energy savings the model would achieve over years of deployment on mobile devices.

The Fine-Tuning Tax: As previously discussed, maintaining accuracy requires extensive retraining. When a model is pruned iteratively, the network is essentially “re-trained” multiple times. From a life cycle assessment perspective, if the energy required to retrain the model exceeds the cumulative energy saved during its operational lifespan on edge devices, the pruning process results in a net negative environmental impact.

This highlights the need to develop “Green Pruning” algorithms that prioritize efficiency during the optimization phase, such as one-shot or data-free methods, rather than focusing solely on the efficiency of the final, static model.

0.1.5 Research Motivation

To address the requirements for efficient compression of convolutional neural networks, it is essential to synthesize a solution that comprehensively resolves the previously discussed challenges. Such a framework must reconcile the dual objectives of maximizing architectural sparsity through the strategic elimination of redundant filters while rigorously preserving the model's representational accuracy. To ensure practical scalability, the designed pruning methodologies should prioritize algorithmic simplicity, thereby maintaining low computational complexity throughout the optimization process. Furthermore, the integration of sustainability as a core design principle is critical to minimize the total energy expenditure during both the pruning and retraining lifecycles. Finally, the proposed approaches must be supported by formal mathematical justifications to ensure the theoretical soundness and interpretability of the filter-pruning procedure.

The core motivations of this research are grounded in the following objectives:

- **Optimization of the Accuracy-Sparsity Trade-off:** A primary motivation of this work is to navigate the inherent tension between aggressive model reduction and the preservation of predictive performance. While standard compression techniques can drastically reduce the parameter count of convolutional neural networks, they often trigger a non-linear degradation in accuracy once a critical sparsity threshold is reached. This research seeks to develop more sophisticated selection metrics that can better identify the "minimum viable architecture", ensuring that the pruned network retains its high-dimensional feature representation capabilities while satisfying the memory constraints of edge hardware.
- **Algorithmic Simplicity and Computational Efficiency:** Current state-of-the-art pruning frameworks often rely on complex, multi-stage pipelines that are difficult to implement and computationally expensive to execute. This complexity acts as a barrier to the adoption of pruning in dynamic environments where models must be adapted quickly. By prioritizing

algorithmic simplicity, this thesis aims to reduce both the software and hardware overhead associated with the pruning process. Developing streamlined algorithms that do not require exhaustive hyperparameter tuning or specialized hardware accelerators is essential for making model compression a more accessible and practical tool for real-world engineering applications.

- **Sustainability and the "Green AI" Paradigm:** In alignment with the growing necessity for sustainable technology, this research is motivated by the need to minimize the total energy required in the pruning process of convolutional neural networks. The "Green AI" paradox demonstrates that the energy consumed during the thousands of GPU hours required for iterative pruning can often outweigh the energy saved during the model's inference phase. Central to this challenge, is the current lack of standardized metrics to evaluate the overhead of the optimization process itself. Therefore, this thesis is motivated by the critical need to define and integrate precise measures of resource efficiency. Establishing these metrics will ensure that the carbon footprint of the development process does not negate the environmental benefits of the resulting compressed model, shifting the focus toward truly "energy-aware" pruning strategies.
- **Mathematical Justification and Theoretical Soundness:** Beyond empirical performance, there is a critical need for a rigorous theoretical foundation for filter removal. Many existing pruning heuristics are based on intuitive but mathematically informal process, which can lead to unpredictable behavior across different network topologies. This research is motivated by the requirement for formal mathematical justifications, utilizing concepts from optimization theory and matrix analysis**,** to provide a principled explanation for filter redundancy. Establishing this theoretical grounding not only ensures the reliability of the pruning procedure but also enhances the interpretability of why certain architectural components are discarded while others are preserved.

0.2 Research Questions

To address the aforementioned challenges and guide our methodology, which will be elaborated in Section 0.3, we formulate three research questions.

0.2.1 Research Question RQ-1

How can a structural pruning framework be theoretically grounded in the intrinsic mathematical properties of convolutional operations to establish importance functions that accurately identify redundancy and preserve predictive performance?

This research addresses the fundamental tension in neural network compression, the trade-off between architectural sparsity and network performance. While the primary challenge lies in identifying a sub-network capable of maintaining the performance of its dense counterpart, current literature remains divided between two divergent perspectives. Magnitude-based methods, such as those proposed by (;and Hans Peter Graf, 2017), prioritize algorithmic simplicity by using filter norms as a proxy for importance. Conversely, more complicated frameworks, including (Luo *et al.*, 2018), (Lin *et al.*, 2020), and (Yvinec, Dapogny, Cord & Bailly, 2023), prioritize accuracy preservation, often at the cost of an excessive computational overhead.

The core limitation of these contemporary approaches is that they are frequently detached from a rigorous mathematical study of the network's underlying representational dynamics. Many pruning heuristics rely on intuitive metrics that lack a solid theoretical anchor. This research bridges the gap by conducting a formal mathematical analysis of the convolutional layer to identify the structural and functional properties that govern feature importance.

0.2.2 Research Question RQ-2

How does the filter selection algorithm influence the efficiency of the pruning process, and can a framework be designed to achieve strong predictive performance with minimal computational complexity?

The operational efficiency of any pruning methodology is fundamentally determined by the design of its filter-selection mechanism and the extent to which it depends on external training data. Within contemporary literature, selection strategies generally fall into two principal categories: *greedy* and *oblivious (non-greedy)* algorithms. Greedy approaches often provide stronger theoretical guarantees through iterative optimization; however, these benefits come at the cost of a substantial computational cost, resulting from repeated evaluation cycles and high algorithmic complexity. In contrast, oblivious strategies offer significant improvements in practical efficiency and scalability, yet frequently exhibit reduced robustness when filters are evaluated independently.

This challenge is further intensified by the growing demand for data-free pruning methodologies motivated by concerns related to data privacy, security constraints, and the memory overhead associated with on-device optimization. Existing data-independent approaches typically assess filters in isolation, overlooking the structural dependencies inherent to deep convolutional architectures, where the output distribution of one layer directly conditions the input representation of subsequent layers.

To address this limitation, the present research introduces a mathematically grounded framework that explicitly models connectivity and signal propagation across adjacent layers. By characterizing these cross-layer interactions without reliance on datasets, the proposed approach identifies redundancy through analysis of the network's internal information transfer mechanisms rather than external supervision.

This interdependency-aware formulation enables the development of a comprehensive data-free pruning strategy that preserves the computational advantages of oblivious selection while mitigating the performance degradation commonly associated with data-independent methods. Ultimately, this research aims to reconcile the competing objectives of speed and accuracy, achieving high compression ratios through a methodology that is computationally efficient and entirely data-agnostic.

0.2.3 Research Question RQ-3

How can a pruning framework be designed to prioritize environmental sustainability, and what metrics are required to rigorously evaluate and quantify the resource efficiency of the pruning process?

A significant limitation within contemporary deep learning literature is the absence of standardized methodologies for assessing the environmental impact and computational overhead associated with the pruning process itself. While existing studies predominantly emphasize the efficiency of the final deployed model, the "Green AI" paradox, namely, the imbalance between deployment savings and optimization cost, remains insufficiently explored. Current state-of-the-art pruning frameworks rarely provide a comprehensive lifecycle analysis of the energy expenditure incurred during filter selection procedure.

Accordingly, there is a clear need to establish formal benchmarks capable of evaluating the sustainability of pruning methodologies. This research proposes a sustainability-oriented pruning architecture supported by a suite of specialized evaluation metric, namely Pruning Resource Efficiency (RE), designed to quantify computational expenditure to obtain a certain amount of accuracy. By formalizing this measure, the proposed framework enables a quantitative assessment method ensuring that the energy savings achieved during long-term deployment are not outweighed by the computational cost required for model compression and retraining.

0.3 Outline of the thesis

This introductory chapter presents the foundational concepts of the thesis, offers an overview of the general context, and defines the problem statement. Chapter 1 reviews the existing literature relevant to the research problems, identifying the gaps in current model compression techniques. Chapter 2 describes the global methodology developed to address the research questions and objectives of the thesis.

The structure of these chapters is outlined as follows:

- **Chapter 3:** ACLI: A CNN Pruning Framework Leveraging Adjacent Convolutional Layer Interdependence and γ-Weakly Submodularity.
- **Chapter 4:** Resource-Efficient and Layer Interdependence-Aware CNN Pruning Leveraging Filter Replacement.
- **Chapter 5:** A Low-Complexity Modified ThiNet Algorithm for Pruning Convolutional Neural Networks.
- **Chapter 6:** Provides a detailed discussion of the core research concepts, focusing on the advantages and limitations of the developed frameworks.
- **Chapter 7:** Provides a final summary of the work's contributions and discusses new directions for future research projects.

CHAPTER 1

LITERATURE REVIEW

1.1 Architectural Compression and the Paradox of Data Dependency

The proliferation of deep Convolutional Neural Networks in mobile and edge computing has catalyzed a transition from purely accuracy-driven research toward hardware-aware optimization. However, the practical feasibility of pruning methodologies is fundamentally governed by their reliance on external data and the computational complexity of their selection mechanisms. Within contemporary deep learning research, pruning strategies are commonly categorized according to their data requirements, each presenting distinct challenges for real-world deployment.

On one hand, data-agnostic or "data-free" approaches, such as those introduced by (;and Hans Peter Graf, 2017), (He, Liu, Wang, Hu & Yang, 2019b), (Lin *et al.*, 2021), and (Chen *et al.*, 2024), aim to achieve efficient model compression by leveraging intrinsic statistics of pre-trained weights. These techniques typically employ l_p-norms, geometric medians, or spectral characteristics to identify architectural redundancy without access to the original training data. While such methods provide strong scalability and address privacy and memory limitations associated with on-device pruning, they have historically struggled to match the representational accuracy achieved by data-driven frameworks. For example, although (;and Hans Peter Graf, 2017) popularized the use of l_1-norms due to their computational simplicity, subsequent analyses have demonstrated that magnitude-based metrics often serve as imperfect indicators of a filter's functional contribution to the pruned network performance. Even advanced data-free extensions such as RED and RED++ (Yvinec, Dapogny, Cord & Bailly, 2021; Yvinec *et al.*, 2023), which incorporate multi-stage clustering and inter-filter correlation analysis, introduce substantial computational overhead through iterative evaluation procedures. This increased complexity frequently diminishes the efficiency advantages associated with data-independent selection, motivating the development of streamlined single-pass pruning strategies.

Conversely, data-driven methods, including (Ganjdanesh, Gao & Huang, 2024), (Li *et al.*, 2023), (Park *et al.*, 2024), and activation-rank analysis approaches such as HRank (Lin *et al.*, 2020), achieve state-of-the-art (SOTA) accuracy by leveraging the informative signal provided by training datasets. This category includes seminal frameworks such as ThiNet (Luo *et al.*, 2018), which optimize filter selection through feature map reconstruction objectives. However, the requirement for representative datasets imposes a significant complexity that limits practical deployment in edge environments. Storing large-scale datasets such as ImageNet on embedded hardware is generally infeasible due to storage and energy constraints, while transmitting data to remote servers introduces latency, bandwidth consumption, and security concerns.

A critical limitation across existing pruning taxonomies is their tendency to evaluate filters in spatial isolation, where importance is assessed independently within each layer. Our proposed method addresses this limitation by explicitly modeling cross-layer parameter interdependence, enabling the pruning process to preserve hierarchical knowledge propagation while remaining entirely independent of external datasets.

1.2 Theoretical Foundations: From Information Theory to Mathematical Bounds

A significant portion of the literature on network pruning is characterized by empirical heuristics that lack a rigorous theoretical foundation. Researchers often rely on "lottery tickets" or intuitive rules of thumb to justify the removal of specific architectural components. To address this lack of rigor, a subset of the research community has turned to information theory and statistical learning theory to provide a principled justification for compression. Foundational work using the Information Bottleneck (IB) principle (Naftali, C & William, 2000), (Tishby & Zaslavsky, 2015) has been explored by (Sarvani, Ghorai, Dubey & Basha, 2022), (Guo *et al.*, 2023), and (Zheng *et al.*, 2021) to formulate pruning as a formal optimization problem. The goal here is to maximize the compression of input information while simultaneously preserving the relevance of the hidden representations to the final output label. While theoretically elegant, the estimation of Mutual Information (MI) in the high-dimensional, non-linear tensor spaces typical of modern CNNs is computationally intractable. Most MI-based methods must rely on Gaussian

approximations or variational bounds, which can lead to sub-optimal pruning decisions. Even efforts to simplify this through the Hilbert-Schmidt Independence Criterion (HSIC) lasso (Guo *et al.*, 2023) remain computationally demanding.

Other theoretical avenues focus on defining mathematical bounds and divergence metrics that do not require explicit data samples. For instance, (Yvinec *et al.*, 2023) derived an upper bound on weight differences to enable efficient filter clustering, effectively treating pruning as a compression problem in the weight space. Building upon this direction, our research moves away from simplistic magnitude-based heuristics toward a more principled mathematical framework. We define an importance function based on a rigorous upper bound of the Average Absolute Error (AAE) in the output of adjacent layers. By proving the γ-weak property of this function, we establish a formal convergence guarantee for the selection process. This provides the mathematical support that is often missing from contemporary “rule of thumb” pruning metrics, ensuring that the resulting sparse architecture is not just empirically successful but theoretically sound.

1.3 Algorithmic Paradigms: The Greedy vs. Oblivious Conflict

The efficiency of a pruning framework is not only a product of its importance metric but also the algorithm used to execute the filter selection. The literature is currently dominated by two algorithmic schools of thought, each representing a trade-off between optimality and speed.

Greedy algorithms are the most prevalent in SOTA methods (Luo *et al.*, 2018; Guo *et al.*, 2023; Lee & Song, 2024) because they provide a structured approach to solving NP-hard optimization problems by making locally optimal choices at each step. In the context of pruning, a greedy algorithm typically removes one filter at a time and then re-evaluates the entire network to determine the next "least important" filter. While these iterative processes offer strong convergence justifications and often result in the highest possible accuracy for a given sparsity level, they are inherently sequential. This "one-at-a-time" evaluation is computationally expensive, requiring hundreds or thousands of forward passes. For researchers and engineers

working in resource-constrained environments, this high "search cost" is a significant barrier to entry.

In contrast, oblivious (non-greedy) algorithms, utilized in more practical or “fast” methods like (;and Hans Peter Graf, 2017) and (Lin *et al.*, 2020), perform filter selection in a single, parallelized pass. These methods rank all filters simultaneously based on a static metric and prune them in one shot. While these methods are significantly faster and more hardware-friendly, they are frequently criticized for their weaker mathematical grounding, as they do not account for the dynamic shifts in the feature space that occur as other filters are removed. However, recent empirical studies have challenged the necessity of the greedy search. For example, our work on F-ThiNet (Tofigh, Ahmad & Swamy, 2022) demonstrated that an oblivious strategy can achieve parity with greedy counterparts in terms of accuracy while offering a significant reduction in time complexity. This thesis extends this paradigm by employing an oblivious algorithm supported by both practical evidence and mathematical proofs, ensuring that complexity remains low without sacrificing the reliability of the pruning results.

1.4 The Evolution of Importance Functions and Selection Criteria

The "Importance Function" is the heart of any structured pruning method, as it dictates which architectural features are essential for the network’s predictive power. Over the last decade, these criteria have evolved through various lenses, ranging from parameter-only metrics to complex joint-optimization formulations.

Weight-dependent approaches offer the greatest simplicity. The norm-based selection popularized by (;and Hans Peter Graf, 2017) remains a baseline in the field, while more recent works like FPGM (He *et al.*, 2019b) argue that filters near the "geometric median" of a layer are redundant because their information is already captured by others. However, these methods are often "blind" to the actual data flow during inference. To bridge this gap, activation-based methods like HRank (Lin *et al.*, 2020), CHIP (Sui *et al.*, 2021), and NISP (Yu *et al.*, 2018) utilize statistics obtained during forward propagation. By measuring the rank of feature maps or the sensitivity

of the final loss to specific activations, these methods can identify filters that are active but irrelevant to the final classification. While more accurate, these methods re-introduce the data dependency mentioned previously, making them difficult to use in privacy-sensitive applications.

A third branch of the literature focuses on regularization-based approaches (Liu *et al.*, 2017), (You, Yan, Ye, Ma & Wang, 2019), (Lin *et al.*, 2019), (Huang & Wang, 2018). These methods do not "search" for a sparse model; instead, they force the network to become sparse during the training phase by applying penalties (such as l_1 or l_2 regularizers) to batch normalization scaling factors or specialized "gate" parameters. While effective, these methods require the network to be trained from scratch or undergo extensive retraining, which is energy-intensive and time-consuming. Finally, optimization-based frameworks (Joo, Yi, Baek & Kim, 2021), (Peng, Wu, Chen & Huang, 2019), (Wang, Grosse, Fidler & Zhang, 2019) attempt to approximate the loss function's sensitivity to filter removal through second-order information, such as the Hessian matrix. While highly precise, calculating the Hessian for millions of parameters is computationally prohibitive.

Despite the diversity of these approaches, there remains a critical need for a weight-dependent criterion that effectively captures inter-layer dependencies without the computational cost of second-order optimization or the data requirements of activation-based methods. Our proposed importance function, synthesized from the error-minimization principles of ThiNet (Luo *et al.*, 2018) and the structural analysis of RED++ (Yvinec *et al.*, 2023), provides a solution. By minimizing a mathematical upper bound of the reconstruction error across adjacent layers, we offer a theoretically grounded, computationally efficient, and data-free criterion that addresses the core limitations of existing structured pruning literature.

A comparative summary of the SOTA pruning methods is given in Table 1.1.

[1] Mathematically Supported

Table 1.1 Comparative analysis of structural pruning methodologies across different characteristics

	Data-Free	**Filter Norm**	**Current Layer**	**Adjacent Layer**	**MS**[1]	**Algorithm**
L1 (;and Hans Peter Graf, 2017)	√	√	√			Oblivious
NORTON (Zniyed, Nguyen *et al.*, 2025)	√		√			Greedy
RED++ (Yvinec *et al.*, 2023)	√		√		√	-
FPGM (He *et al.*, 2019b)	√		√			-
HRank (Lin *et al.*, 2020)				√		-
ThiNet (Luo *et al.*, 2018)				√		Greedy
F-ThiNet (Tofigh *et al.*, 2022)				√		Oblivious
IPBM (Guo *et al.*, 2023)			√		√	Greedy
OUR method	√	√	√	√	√	Oblivious

CHAPTER 2

OBJECTIVES AND GENERAL METHODOLOGY

This chapter provides a comprehensive articulation of the thesis objectives, systematically addressing the identified research problems and questions while situating them within the context of current literature limitations. To enhance clarity and facilitate a deeper comprehension of the proposed methodology, we illustrate the conceptual and procedural relationships between these research phases. This organizational framework serves as a roadmap for the thesis structure, ensuring a logical progression from theoretical inquiry to practical implementation.

2.1 Main Hypothesis

The research hypothesis (RH) of this framework is defined as follows:

RH: The central hypothesis of this research is that by mathematically formalizing the inter-layer dependencies between consecutive convolutional layers and identifying the critical factors governing filter pruning impact on subsequent activations, it is possible to derive a data-free importance function that achieves high performance with minimal computational complexity. Furthermore, it is posited that by integrating this importance function within an oblivious (non-greedy) selection framework and evaluating it against a novel Resource Efficiency metric, a pruning paradigm can be established that simultaneously optimizes both model performance and environmental sustainability without the typical trade-offs associated with data-agnostic compression.

2.2 Main Objective

The primary goal of this doctoral research is to establish a principled framework for the compression of Deep Convolutional Neural Networks (CNNs) that harmonizes architectural efficiency with environmental sustainability. The main objective (MO) is formally defined as follows:

MO: To develop a mathematically grounded, data-free pruning strategy that effectively identifies and eliminates redundant convolutional filters by modeling the structural interdependencies between adjacent layers. This strategy aims to achieve a superior trade-off between the pruned network's size and its representational accuracy, ensuring performance levels that are comparable to or exceed current state-of-the-art (SOTA) data-driven methods. Furthermore, this objective includes the formulation of a novel Resource Efficiency metric to quantify and minimize the energy-computational footprint of the pruning lifecycle, thereby fostering a "Green AI" paradigm in model optimization.

2.2.1 Specific Objectives

The main objective (MO) is divided into three specific objectives (SO-1 to SO-3), each designed to systematically address the research questions (RQ-1 to RQ-3) identified in Section 0.2:

2.2.1.1 Specific Objective SO-1

SO-1: Establishing a mathematically grounded framework for structural model optimization to enable a high-performance convolutional neural network pruning.

The objective of this research is to develop a unified empirical and mathematical framework for analyzing, identifying, and restructuring critical filters within convolutional neural networks while rigorously preserving predictive accuracy and representational integrity. This work first seeks to systematically characterize the identifying properties of convolutional filters that contribute most significantly to feature extraction and classification performance. Through the comparative evaluation of multiple filter-ranking criteria, the study aims to detect architectural redundancy across different network depths and to construct detailed sensitivity profiles that quantify how modifications at early, intermediate, and late layers reshape the global loss landscape.

Beyond empirical assessment, this objective integrates a formal mathematical investigation into the dynamics of signal propagation between adjacent convolutional layers. Specifically, analytical

bounds on approximation error, measured through Average Absolute Error, will be derived to quantify how filter removal or substitution in layer L perturbs the feature representations of layer $L + 1$. By modeling the cascading effects of structural sparsity on the network's internal feature manifold, the research establishes a principled justification for pruning decisions grounded in the algebraic structure of convolutional weight tensors rather than purely stochastic heuristics.

A central component of this framework is the exploration of non-zero filter replacement strategies as an alternative to conventional elimination. The research will investigate lower-complexity substitutes capable of preserving high-frequency representations and minimizing catastrophic forgetting, thereby explicitly addressing the non-linear trade-off between architectural sparsity, quantified through parameter count and FLOPs reduction, and representational accuracy, evaluated using the Top-1 error metric. Mathematical conditions ensuring the stability, convergence, and near-optimality of the selection and pruning algorithm will also be formally established, providing theoretical guarantees for predictable performance.

Ultimately, this objective aims to bridge empirical model compression and optimization theory by defining a robust, theoretically grounded selection framework that determines optimal redundancy thresholds and validates filter replacement as a computationally efficient yet accuracy-preserving paradigm for convolutional network reduction.

2.2.1.2 Specific Objective SO-2

SO-2: Designing a low-complexity filter selection architecture for efficient and data-free convolutional neural network pruning

The objective of this research is to investigate how filter selection architectures influence the computational efficiency and practical scalability of neural network pruning, to design a high-performance framework capable of achieving substantial model compression under minimal computational complexity. This work addresses the trade-off between selection optimality and execution efficiency by conducting a rigorous analysis of the computational and temporal overhead inherent in modern pruning strategies, with particular emphasis on

the divergence between iterative greedy mechanisms and oblivious selection paradigms. By systematically examining the "complexity–performance gap," the study evaluates how context-aware greedy approaches, despite their strong empirical performance, introduce significant hardware bottlenecks through repeated inference cycles and weight re-evaluation, limiting deployment in real-time and resource-constrained environments.

Central to this objective is the development of a single-pass filter selection architecture grounded in an oblivious algorithmic framework. Unlike sequential greedy strategies that require continual recomputation to adapt to parameter updates, the proposed methodology performs global filter ranking through a non-iterative process designed to operate in near-constant time. The research will analyze how such a streamlined selection mechanism preserves structural coherence and pruning effectiveness while substantially reducing optimization overhead. Efficiency gains will be quantified through reductions in FLOPs and execution time during the pruning phase, demonstrating that computational simplicity does not necessarily imply performance degradation.

To further enhance deployability, this objective also investigates pruning under strict data-independence constraints. Specifically, the framework aims to eliminate reliance on access to original training datasets by leveraging intrinsic structural properties and inter-layer dependencies as proxies for data-driven importance signals. By modeling the internal "knowledge flow" of the network, the research seeks to demonstrate that accurate filter importance estimation and high-quality compression can be achieved in a data-free environment. Ultimately, this objective aims to establish a theoretically informed and computationally efficient pruning architecture capable of delivering competitive performance while minimizing algorithmic complexity, enabling scalable optimization for edge-device and privacy-constrained applications.

2.2.1.3 Specific Objective SO-3

SO-3: Developing a lifecycle-aware quantitative framework for evaluating the environmental and computational sustainability of convolutional neural network pruning.

The third objective is to establish a rigorous quantitative framework for assessing the environmental and computational sustainability of neural network optimization, shifting the paradigm from static model efficiency to lifecycle-aware compression. This involves the systematic derivation of the *Resource Efficiency* measure. Unlike traditional metrics that focus exclusively on the inference cost of the final architecture, this framework will account for the cumulative energy consumption during the filter selection phase and the iterative search cycles. By analyzing the trade-off between the computational complexity of the network optimization and the resulting accuracy of the pruned network, this objective seeks to define a measure for "sustainable pruning."

2.3 General Methodology

To systematically address the research questions (RQ-1 to RQ-3) and achieve the specific objectives (SO-1 to SO-3) established in the preceding sections, this thesis proposes a cohesive framework comprising three interconnected methodologies, denoted as M1 through M3. These methodologies are structured to transition from the foundational mathematical modeling of the network to the engineering of sustainable, data-free optimization strategies. The specific methodologies are defined as follows:

2.3.1 Methodology M1:

This methodology establishes a unified framework that transitions from the rigorous mathematical investigation of signal dynamics to the empirical characterization of filter importance across hierarchical architectures. By grounding structural optimization in the intrinsic algebraic properties of weight tensors, this phase ensures that model reduction is both theoretically sound and empirically robust.

The key components of this integrated methodology are as follows:

- Derivation of Mathematically Grounded Saliency Metrics: This component focuses on the formulation of a formal importance function rooted in the modeling of structural

interdependencies between adjacent convolutional layers. Unlike traditional heuristics, this function is derived by establishing rigorous upper bounds for the Average Absolute Error propagated to subsequent feature maps. This task provides a principled mathematical relationship between structural modifications in layer L and the resulting representational deviation in layer L+1.

- Optimal Non-Zero Filter Replacement Paradigm: Moving beyond simple filter elimination, this task utilizes matrix analysis to identify optimal non-zero replacements for redundant parameters. By deriving simplified, lower-dimensional filter alternatives that minimize reconstruction error, this framework ensures that essential knowledge flow is preserved. This approach allows for a reduction in computational expenditure while maintaining a high network performance.
- Formal Verification of Algorithmic Stability and Convergence: A core theoretical contribution of this methodology is the formal proof of the γ-weak property for the proposed importance function. This task provides a mathematical guarantee for the convergence of the oblivious (non-greedy) selection algorithm, demonstrating that a single-pass ranking strategy remains stable and reliable. This research bridges the gap between empirical structural compression and optimization theory.
- Multi-Level Performance and Complexity Analysis: The final component involves a systematic investigation of the network's representational fidelity across a spectrum of pruning scales. Performance is quantified using Top-1 and Top-5 accuracy to measure the performance of the pruned network. Simultaneously, an analytical evaluation of the algorithmic overhead, quantified through FLOPs and multiplication counts during the selection phase, is conducted to rigorously assess the resource efficiency and sustainability of the optimization procedure.

2.3.2 Methodology M2:

This integrated methodology bridges the gap between high-efficiency algorithmic engineering and large-scale empirical validation. By transitioning from computationally expensive iterative

selection to a streamlined, oblivious paradigm, this phase focuses on creating a robust, data-free optimization pipeline capable of high-throughput execution across diverse architectural benchmarks.

The key components of this integrated methodology are:

- Engineering of a High-Throughput Oblivious Selection Pipeline: This component focuses on resolving the inherent computational bottlenecks of traditional greedy pruning. By investigating the divergence between greedy and oblivious (non-greedy) strategies in SOTA methods, this task develops a single-pass selection logic. A primary case study involves re-engineering established frameworks, such as ThiNet, to quantify the specific trade-offs between representational accuracy and the significant reduction in temporal overhead required for real-time applications.
- Validation of the Data-Free Paradigm and Internal "Knowledge Flow": A core focus of this phase is the empirical proof of a truly data-free compression framework. By utilizing only the intrinsic weight tensors and the inter-layer dependencies derived in previous theoretical phases, this component demonstrates that high-performance pruning is achievable without access to original training distributions. This effectively addresses critical constraints related to data privacy, security, and the memory limitations of on-device deployment.
- Architectural Scalability and Universal Benchmarking: To ensure the framework's universality, the proposed algorithms are subjected to a rigorous evaluation across convolutional architectures of varying depth and complexity, including VGG, ResNet, and MobileNet. This task validates the framework's ability to model interdependencies in both residual and non-residual structures, ensuring high performance across both high-capacity and mobile-optimized models.
- Cross-Dataset Fidelity Assessment: The robustness of the representational fidelity is validated across datasets of varying scale, such as CIFAR-10 and ImageNet. This allows for a comprehensive characterization of how the importance functions and replacement strategies perform across different categories of visual information, ensuring that the results remain consistent and unbiased regardless of data complexity or scale.

2.3.3 Methodology M3:

The third methodology establishes a "Green AI" evaluation protocol by deriving a rigorous quantitative framework for assessing the environmental and computational sustainability of the pruning process. This phase shifts the focus from purely inference-based metrics to a comprehensive process analysis of the optimization process. The key components of Methodology M3 are as follows:

- The derivation of the RE metric, defined as the total cumulative number of FLOPs and multiplications required throughout the entire pruning procedure to achieve a target representational accuracy. Unlike static metrics, RE provides a dynamic assessment of the energy "cost-to-benefit" ratio.
- A systematic benchmarking of the proposed methodology against state-of-the-art (SOTA) pruning frameworks. This component evaluates the efficiency gains of the oblivious selection paradigm by quantifying the reduction in computational overhead required to reach standardized accuracy thresholds across varying network architectures.
- The formulation of a standardized mathematical model to compare the carbon footprint of neural network pruning methods. This task focuses on developing a hardware-agnostic formula that translates computational complexity (FLOPs) and execution time into CO_2 , facilitating a universal comparison of the environmental impact of compression algorithms regardless of the underlying silicon architecture.

A summary diagram of the thesis is presented in Figure 2.1.

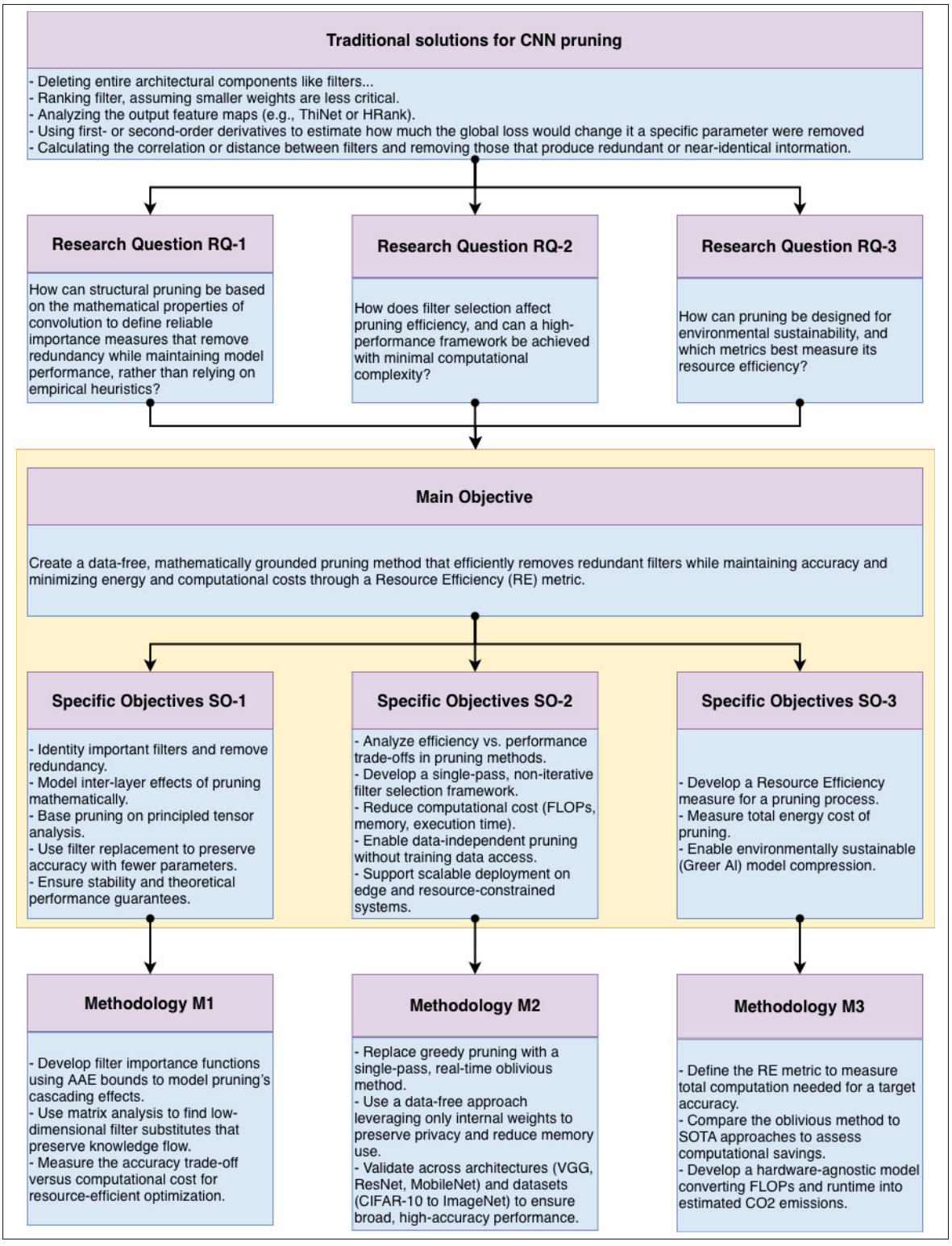

Figure 2.1 Outline diagram of the thesis

CHAPTER 3

ACLI: A CNN PRUNING FRAMEWORK LEVERAGING ADJACENT CONVOLUTIONAL LAYER INTERDEPENDENCE AND γ-WEAKLY SUBMODULARITY

Sadegh Tofigh[1] , Mohammad Askarizadeh [1] , M. Omair Ahmad [2] , M. N. S. Swamy [2] , Kim Khoa Nguyen [1]

[1] Department of Electrical Engineering, École de Technologie Supérieur, 1100 Notre-Dame Ouest, Montréal, Québec, Canada H3C 1K3

[2] Department Electrical and Computer Engineering, Concordia University, 1455 Blvd. De Maisonneuve Ouest, Montreal, Quebec, Canada H3G 1M8

Article accepted by IEEE Transactions on Pattern Analysis and Machine Intelligence, September 2025.

Abstract

Today, convolutional neural network (CNN) pruning techniques often rely on manually crafted importance criteria and pruning structures. Due to their heuristic nature, these methods may lack generality, and their performance is not guaranteed. In this paper, we propose a theoretical framework to address this challenge by leveraging the concept of γ-weak submodularity, based on a new efficient importance function. By deriving an upper bound on the absolute error in the layer subsequent to the pruned layer, we formulate the importance function as a γ-weakly submodular function. This formulation enables the development of an easy-to-implement, low-complexity, and data-free oblivious algorithm for selecting filters to be removed from a convolutional layer. Extensive experiments show that our method outperforms state-of-the-art benchmark networks across various datasets, with a computational cost comparable to the simplest pruning techniques, such as l_2-norm pruning. Notably, the proposed method achieves an accuracy of 76.52%, compared to 75.15% for the overall best baseline, with a 25.5% reduction in network parameters. According to our proposed resource-efficiency metric for pruning methods, the ACLI approach demonstrates orders-of-magnitude higher efficiency than the other baselines, while maintaining competitive accuracy.

Keywords: Deep learning, pruning, data-free, machine learning, convolutional neural networks, model comparison.

3.1 Introduction

Convolutional neural networks (CNNs) have demonstrated significant success across various computer vision tasks, including image classification (Szegedy *et al.*, 2015), (He, Zhang, Ren & Sun, 2016b), object detection (Girshick, Donahue, Darrell & Malik, 2014), (Ren, He, Girshick & Sun, 2015), and image segmentation (Long, Shelhamer & Darrell, 2015), (Chen, Papandreou, Kokkinos, Murphy & Yuille, 2017). However, the ever-increasing demands on computing power and memory footprint present challenges for deploying CNNs on edge devices. As a result, various model compression techniques have been proposed to compress and accelerate networks, e.g., pruning, parameter quantization, and knowledge distillation (Cheng, Wang, Zhou & Zhang, 2017). Among them, pruning methods have been recognized as effective tools to facilitate model deployment by eliminating redundant filters in CNNs. A pruning method is primarily characterized by two key components: the parameters (e.g., filters) in the convolutional layers of CNNs and the input dataset used by the network (Cheng *et al.*, 2017). Regarding parameters, existing pruning methods can be categorized into two main groups: *structured* pruning methods (Lin *et al.*, 2020), (;and Hans Peter Graf, 2017), (Luo *et al.*, 2018), (Alwani, Madhavan & Wang, 2021), (Lin *et al.*, 2021) and *unstructured* pruning methods (Ding *et al.*, 2019), (Frankle & Carbin, 2019), (Han, Pool, Tran & Dally, 2015). Unlike models pruned with unstructured methods, which often require specialized hardware, structured pruning methods are typically favored because the resulting pruned models can operate efficiently with standard hardware.

Pruning methods can be broadly classified into two main categories based on their use of datasets. The first category, known as *data-based* methods, involves methods that select filters from convolutional layers by analyzing their output, leveraging the dataset during the pruning process (El Halabi, Srinivas & Lacoste-Julien, 2022), (Luo *et al.*, 2018), (Tofigh *et al.*, 2022), (Lin *et al.*, 2020), (Lin *et al.*, 2019). These data-based approaches select filters for pruning by either

examining the features from the *current layer* or those from the *adjacent layer* (He & Xiao, 2024). While current layer methods make their decisions based on the features of the pruned layer, adjacent layer approaches consider all removed parameters, including those from the adjacent layer. Although adjacent layer pruning methods offer a more comprehensive assessment of parameter importance by considering adjacent layer during filter selection, these approaches often increase the computational complexity.

Table 3.1 Summary of the existing pruning methods and ours

	Data-Free	**Filter Norm**	**Current Layer**	**Adjacent Layer**	**Algorithm**
L1 (;and Hans Peter Graf, 2017)	√	√	√		Oblivious
ACLI (OURS)	√	√	√	√	Oblivious
RED++ (Yvinec *et al.*, 2023)	√		√		-
FPGM (He *et al.*, 2019b)	√		√		-
HRank (Lin *et al.*, 2020)				√	-
ThiNet (Luo *et al.*, 2018)				√	Greedy
F-ThiNet (Tofigh *et al.*, 2022)				√	Oblivious
IPBM (Guo *et al.*, 2023)			√		Greedy

The second category consists of *data-free* methods (;and Hans Peter Graf, 2017), (He *et al.*, 2019b), (Yvinec *et al.*, 2021), (Yvinec *et al.*, 2023), where filter selection for pruning is based solely on the parameters within the convolutional layers, without leveraging any dataset information. Data-free pruning methods are further divided into two subgroups: *filter norm* and *filter correlation* (He & Xiao, 2024). While methods in the filter norm group offer significantly lower computational complexity, they generally result in less effective accuracy in the pruned network compared to those in the filter correlation group. In general, while data-based pruning methods tend to exhibit better performance than data-free methods, the latter methods offer significantly lower complexity in filter selection compared to the former. No matter whether the pruning method is structured or unstructured, and whether it is data-free or data-based, most of these proposed methods are heuristics that offer no theoretical guarantees (Luo *et al.*, 2018), (Lin *et al.*, 2020), (;and Hans Peter Graf, 2017), (He *et al.*, 2019b). Moreover, the complexity of these pruning methods, in terms of the number of operations required, imposes significant challenges and makes these methods less accessible to practitioners. Table 3.1 shows the characteristics of state-of-the-arts pruning methods and compares them based on their respective categories.

In this paper, we propose a structured, easy-to-implement, and data-free pruning method based on a novel approach to filter selection. Unlike existing pruning methods whose *importance functions* are primarily designed based on empirical research, our proposed importance function is supported by a robust theoretical framework. For example, although our method builds also on the filter selection strategy of ThiNet (Luo *et al.*, 2018), which prunes filters based on the absolute error in the $(l+1)^{st}$ layer output, we address its computational complexity by introducing an upper bound on the *average absolute error* (AAE) in the next layer's output. This novel upper bound allows us to design an importance function that efficiently selects filters for pruning from the l^{th} layer while ensuring that the induced error by this removal in the adjacent $(l+1)^{st}$ layer remains controlled. By doing so, unlike traditional data-free pruning methods, our proposed *Adjacent Convolutional Layer Interdependence* (ACLI) pruning method is powered by a comprehensive adjacent layer pruning approach. This approach analyzes the interdependence between the layer to be filtered and its adjacent layers, resulting in a more informative filter selection process, as illustrated in Figure 3.1. To address the computational burden of the *greedy* algorithms for pruning (e.g., (El Halabi *et al.*, 2022), (Luo *et al.*, 2018)), which is not affordable for resource-constrained devices, our filter selection employs a low-complexity *oblivious* algorithm. This is made possible by the γ-weakly submodular characteristic of our pruning optimization problem. According to Table 3.1, the method most similar to ACLI in

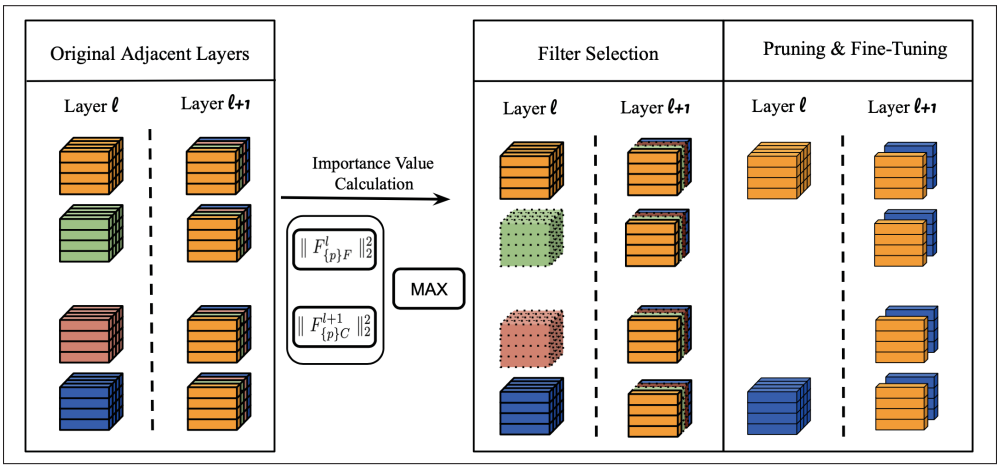

Figure 3.1 Framework of the ACLI pruning method

terms of the properties of its importance function is L1 (;and Hans Peter Graf, 2017). However, unlike L1, ACLI also considers the parameters of the adjacent layer, making it more informative. Finally, our extensive experiments demonstrate that the proposed ACLI method significantly outperforms existing filter pruning techniques. It achieves this by maintaining accuracy while matching the simplicity of filter norm methods.

In summary, our main contributions are as follows:

- We develop an importance function based on an upper bound of the absolute error in the output of the layer succeeding the pruned layer, effectively minimizing the impact on network accuracy.
- Our importance function is derived solely from the norm of the filters in convolutional layers, thereby significantly reducing the complexity of the pruning process to a level comparable with filter norm Methods.
- We utilize a low-complexity and rapidly converging oblivious algorithm instead of a greedy algorithm. The γ-weakly submodular structure of our proposed pruning method provides theoretical support for the convergence of the oblivious algorithm.
- We empirically validate that the proposed pruning method effectively restores the accuracy of the pruned network, achieving results comparable to the most advanced pruning methods currently available.
- We propose a resource efficiency (RE) metric for evaluating pruning methods, which considers both the accuracy of the pruned network and the computational complexity of the pruning process, defined as: $RE = \left(\frac{\text{ACC}}{\text{MAC}}\right)$, i.e., the ratio of network accuracy (ACC) to the number of Multiply-Accumulate Operations (MAC). To the best of our knowledge, so far, this is the first RE metric is ever proposed for pruning methods.

3.2 Related Work

The primary requirement for a structured pruning method is its effectiveness in recovering the accuracy of the pruned network. Additionally, a reliable and practical pruning method should

be supported by a solid mathematical foundation that can easily be implemented through a low-complexity algorithm.

3.2.1 Complexity of Pruning Methods

From both complexity and accuracy perspectives, most methods with low-complexity are not accurate enough to use, while the highly-accurate pruning methods are usually too complex. The complexity of a pruning method is decided by its reliance on data and the algorithm used for filter selection. Prior works (;and Hans Peter Graf, 2017), (He *et al.*, 2019b), (Lin *et al.*, 2021) exclusively prune filters in a data-free manner, suggesting less complex algorithms, their performance in terms of network accuracy often lags behind data-based approaches (Lin *et al.*, 2020), (Luo *et al.*, 2018), (Lin *et al.*, 2019), (Tofigh *et al.*, 2022), (Alwani *et al.*, 2021), (Huang & Wang, 2018), in which dataset is integrated into filter selection alongside network parameters, resulting in a computationally intensive process. Although the pruning methods proposed by (Yvinec *et al.*, 2021) and (Yvinec *et al.*, 2023) are data-free and less complex than their data-based counterparts, they achieve commendable performance. However, their three-step process introduces a level of complexity higher than what is typically expected from a data-free approach. According to the classification by (He & Xiao, 2024), data-free structured pruning methods are categorized into filter norm and filter correlation groups. The filter norm methods, exemplified by (;and Hans Peter Graf, 2017), are more efficient in terms of complexity, as it determines filter importance based solely on the norm. On the other hand, (He *et al.*, 2019b), (Yvinec *et al.*, 2023), (Yvinec *et al.*, 2021) fall under the filter correlation category. Notably, all existing data-free pruning methods calculate filter importance based solely on the filters in the current layer, without accounting for the dependencies between these filters and their corresponding channels in adjacent layers. This limitation makes them less comprehensive compared to data-based methods.

In this paper, we propose ACLI, a data-free and filter norm pruning approach. Unlike other methods in this category, ACLI offers a more comprehensive filter selection process by accounting for the interdependence between parameters in the pruned layer and its adjacent layers.

3.2.2 Mathematical Perspective for Pruning

Most of the proposed pruning methods in the literature are heuristic and lack theoretical guidance. Some prior works have explored a theoretical perspective for pruning networks. Studies (Sarvani *et al.*, 2022), (Guo *et al.*, 2023), (Wang, Lu & Blankevoort, 2020), (Dai, Zhu, Guo & Wipf, 2018), (Zheng *et al.*, 2021) investigate the removal of filters based on *mutual information* (MI) or a variation of MI known as the *information bottleneck* (IB) (Naftali *et al.*, 2000), (Tishby & Zaslavsky, 2015). However, the computation of MI and IB involves density estimation in high-dimensional space, which is computationally infeasible in most practical cases. To cope with this challenge, some of these pruning methods propose new ideas, e.g., (Guo *et al.*, 2023) implements its IB pruning problem by solving a *Hilbert-Schmidt independence criterion lasso* problem under certain conditions, but they still have a high complexity filter selection (Guo *et al.*, 2023). Moreover, (Wang *et al.*, 2020), (Dai *et al.*, 2018), (Sarvani *et al.*, 2022) within this category might struggle to recover their performance after pruning, so they use special fine-tuning methods to solve the problem. Similarly, the mathematical framework proposed in (Zheng *et al.*, 2021) is not sufficiently helpful to avoid heuristic pruning criteria like l_1-norm. Notably, (El Halabi *et al.*, 2022) presents a structured pruning method focused on the limited-data regime that relies on reweighting methods (Mariet & Sra, 2016) and considers submodular optimization. Furthermore, (Yvinec *et al.*, 2023) introduces an upper bound on weight differences to quantify parameter similarity, which enables efficient weight clustering for filter pruning.

In this work, we propose an importance function based on an upper bound of the AAE in the output of the adjacent layer. Additionally, the γ-weak characteristic of the proposed importance function guarantees the convergence of the oblivious algorithm used for filter selection.

3.2.3 Greedy and Oblivious Algorithm in Pruning Methods

Greedy algorithms are frequently employed by pruning methods as the means of converging to an optimized solution (Luo *et al.*, 2018), (Guo *et al.*, 2023). While they offer a promising

approach for tackling NP-hard problems, the complexity of greedy algorithms remains high due to their iterative nature, making them less suitable for devices with limited computing resources. To address this, some studies have explored alternative oblivious algorithms, albeit without providing a rigorous mathematical justification (Tofigh *et al.*, 2022), (Lin *et al.*, 2020), (;and Hans Peter Graf, 2017). Although greedy algorithms are grounded in a stronger mathematical foundation, it has been heuristically demonstrated that oblivious algorithms can achieve comparable performance. For instance, in our previous work, F-ThiNet (Tofigh *et al.*, 2022) employs the same filter selection strategy as ThiNet (Luo *et al.*, 2018) but utilizes an oblivious algorithm instead of a greedy one, achieving similar performance with significantly lower complexity. In this work, we employ an oblivious algorithm to develop a low-complexity pruning method tailored for resource-limited devices.

3.2.4 Importance Function in Pruning Methods

Structured pruning methods define their importance functions from various perspectives, focusing on different components of convolutional neural networks (He & Xiao, 2024). Weight-dependent importance functions determine filter selection solely based on the filters in convolutional layers. For example, (;and Hans Peter Graf, 2017) employs filter norms as the basis for importance, while FPGM (He *et al.*, 2019b) and RED++ (Yvinec *et al.*, 2023) use the median and a three-step pruning, respectively, for computing importance. On the other hand, several methods incorporate dataset information into the importance calculation. Both ThiNet (Luo *et al.*, 2018) and F-ThiNet (Tofigh *et al.*, 2022) select filters for pruning by approximating the absolute error in the output of the adjacent layer, whereas HRank (Lin *et al.*, 2020) relies on the rank of the output as a selection metric. Similarly, CHIP (Sui *et al.*, 2021) and NIPS (Yu *et al.*, 2018) assess filter importance based on activation values obtained during forward propagation. Additionally, some pruning techniques employ sparsity regularizers to induce structured sparsity in networks. Methods such as NS (Liu *et al.*, 2017), GBN (You *et al.*, 2019), PR (Zhuang *et al.*, 2020), and RSNLI (Ye, Lu, Lin & Wang, 2018) reduce network size by applying regularizers to batch normalization (BN) parameters. Approaches like SSS (Huang & Wang, 2018) and GAL (Lin *et al.*, 2019) prune

by introducing regularizers on additional parameters defined in their filter selection process. Alternatively, optimization tools can be used to determine filter importance. For instance, (Peng *et al.*, 2019), (Wang *et al.*, 2019), and (Nonnenmacher, Pfeil, Steinwart & Reeb, 2022) apply Taylor expansion to approximate the loss function when filters are set to zero, thereby computing their importance.

In this work, we propose a weight-dependent importance function which is based on the norm of the filters in the pruned and adjacent layers. ACLI importance function, inspired by ThiNet (Luo *et al.*, 2018) and RED++ (Yvinec *et al.*, 2023), controls AAE by minimizing an upper bound on it in the output of the adjacent layer.

3.3 ACLI Approach Overview

In this section, we present the optimization problem of CNN pruning. We introduce an importance function designed to minimize the upper bound on the average absolute error in the output of the layer following the pruned layer. Additionally, we mathematically prove that this importance function is γ-weakly submodular.

3.3.1 Pruning Technique

Selecting k filters out of H_l filters for removal from l^{th} convolutional layer ($l = 1, \ldots, \mathbb{L}$) with the minimum importance is equivalent to the selection of the $H_l - k$ most important filters for keeping in l^{th} layer. Choosing $H_l - k$ filters with highest importance level to retain in l^{th} layer is formulated as the following optimization problem

$$\max_{\substack{S \subset \{1,\ldots,H_l\} \\ |S| = H_l - k}} \mathcal{L}(S), \tag{3.1}$$

where $\mathcal{L}(S)$ indicates the importance of filters indexed by S. Solving this optimization problem is an NP-hard problem, intractable in terms of computational resources and time.

In order to define our ACLI importance function $\hat{\mathcal{L}}$, we investigate the impact of removing P_r^{th} filter $F^l[:,:,:,p_r]$ from the l^{th} convolutional layer on X_{l+1}, the output of the $(l+1)^{st}$ convolutional layer. Given the original output X_{l+1} of $(l+1)^{st}$ convolutional layer, with $*$ as the symbol for convolution-multiplication and $F^{l+1}[:,:,p,:]$ as the p^{th} channel in F^{l+1}, the filters in $(l+1)^{st}$ convolutional layer

$$X_{l+1} = F^{l+1} * X_l = \sum_{p=1}^{H_l} F^{l+1}[:,:,p,:] * X^l[:,:,p], \tag{3.2}$$

and the output $\hat{X}_{l+1}^{p_r}$ of $(l+1)^{st}$ convolutional layer when p_r^{th} filter from l^{th} convolutional layer is pruned

$$\begin{aligned}\hat{X}_{l+1}^{p_r} = &\sum_{p=1}^{p_r-1} F^{l+1}[:,:,p,:] * X^l[:,:,p] \\ &+ \sum_{p=p_r+1}^{H_l} F^{l+1}[:,:,p,:] * X^l[:,:,p],\end{aligned} \tag{3.3}$$

the magnitude of the difference between the aforementioned two outputs is expressed as

$$\begin{aligned}|X_{l+1} - \hat{X}_{l+1}^{p_r}| &= |F^{l+1}[:,:,p_r,:] * X^l[:,:,p_r]| \\ &= |F^{l+1}[:,:,p_r,:] * f\left(F^l[:,:,:,p_r] * X^{l-1}\right)|,\end{aligned} \tag{3.4}$$

where f is a nonlinear function defining the operations performed by the activation function and the pooling layer. Equation (3.4) states that a pruning method that identifies the least important filter (i.e., the filter whose removal causes the smallest change in the network performance) must consider the parameters of both l^{th} and $(l+1)^{st}$ convolutional layers. Therefore, we propose an importance function that adheres to this principle.

3.3.1.1 Data-Free ACLI Importance Function

We define the importance function $\hat{\mathcal{L}}$ on a subset $S \subset \{1, \ldots, H_l\}$ of indices as follows

$$\hat{\mathcal{L}}(S) := \max\{\| F^l[:,:,:,S] \|_2^2, \| F^{l+1}[:,:,S,:] \|_2^2\}, \tag{3.5}$$

where $F^l[:,:,:,S]$ and $F^{l+1}[:,:,S,:]$ represent the tensor created by filters in l^{th} layer with indices in S and tensor of the channels of the filters in $(l+1)^{st}$ layer with indices from S, respectively.

Note: For every set $S \subset V$ of size k, the following inequalities are hold for importance function $\hat{\mathcal{L}}$ defined in (3.5)

$$\hat{\mathcal{L}}(S) \leq \sum_{i \in S} \hat{\mathcal{L}}(i), \qquad \frac{\sum_{i \in S} \hat{\mathcal{L}}(i)}{k} \leq \hat{\mathcal{L}}(S). \tag{3.6}$$

The importance function $\hat{\mathcal{L}}$ is not only designed based on the interdependence between the parameters of the l^{th} and $(l+1)^{st}$ convolutional layers, but it is also formulated to ensure minimal impact on network performance. To demonstrate this minimal impact, the following theorem provides an upper bound on the absolute error in the output of $(l+1)^{st}$ convolutional layer when the p_r^{th} filter is pruned from the l^{th} layer. The proof is provided in the supplementary material.

Theorem 1. *In a CNN, let X_{l+1} be the output of the $(l+1)^{st}$ convolutional layer and $\hat{X}_{l+1}^{p_r}$ be the output when p_r^{th} filter from the l^{th} layer ($l = 1, \ldots, \mathbb{L}$) is pruned. The absolute error in the output of the $(l+1)^{st}$ convolutional layer is bounded above as follows*

$$\| X_{l+1} - \hat{X}_{l+1}^{p_r} \|_2^2 \leq \| F^{l+1}[:,:,p_r,:] \|_2^2 \| F^l[:,:,:,p_r] \|_2^2 \, \Lambda_l, \tag{3.7}$$

where Λ_l depends solely on the parameters of the convolutional layers preceding the l^{th} layer and on the input dataset. For more details on Λ_l, see section 3.7.1 in the supplementary material.

The upper bound presented in Theorem 1 is derived for a specific input data, meaning the absolute error in the output of the $(l+1)^{st}$ convolutional layer varies when using a different input data. To establish an upper bound based on the entire dataset, we compute the average absolute error in the output of the $(l+1)^{st}$ convolutional layer, denoted as $AAE(X_{l+1})$, and expressed as follows:

$$\begin{aligned} AAE(X_{l+1}) &= E(\parallel X_{l+1} - \hat{X}_{l+1}^{p_r} \parallel_2^2) \\ &\leq \parallel F^{l+1}[:,:,p_r,:] \parallel_2^2 \parallel F^{l}[:,:,:,p_r] \parallel_2^2 E(\Lambda_l). \end{aligned} \tag{3.8}$$

The upper bound given by Equation (3.8) explains the rationale behind the importance function $\hat{\mathcal{L}}$ defined in (3.5). The importance function $\hat{\mathcal{L}}$ ensures that when the maximum of the norms $\parallel F^{l+1}[:,:,p_r,:] \parallel_2^2$ and $\parallel F^{l}[:,:,:,p_r] \parallel_2^2$ for the p_r^{th} filter is small, both norms are minimized, thereby reducing the upper bound in Equation (3.8) and ensuring that removing this filter causes the least average absolute error in the output of the $(l+1)^{st}$ convolutional layer.

3.3.1.2 Tightness of the Upper Bound on the AAE

In the upper bound presented in Equation (3.8), which pertains to the average absolute error in the output of the $(l+1)^{st}$ convolutional layer, the removal of the p_r^{th} filter in the l^{th} layer is justifiable if the upper bound is sufficiently tight. To demonstrate the tightness of this bound, we simplify the analysis by considering the equivalent inequality

$$\frac{AAE(X_{l+1})}{E(\Lambda_l)} \leq \parallel F^{l+1}[:,:,p_r,:] \parallel_2^2 \parallel F^{l}[:,:,:,p_r] \parallel_2^2 . \tag{3.9}$$

As aforementioned, the value of $E(\Lambda_l)$ is a scalar that depends on both the layers preceding the l^{th} convolutional layer and the dataset used to train the network. However, the upper bound in Equation (3.9) is determined solely by the parameters of the pre-trained network. To evaluate the tightness of the upper bound in Equation (3.8), we observed consistently low values for

$$\parallel F^{l+1}[:,:,p_r,:] \parallel_2^2 \parallel F^{l}[:,:,:,p_r] \parallel_2^2 \ll 1 \tag{3.10}$$

across the three networks studied as examples in this paper, i.e., ResNet-50, ResNet-56, and ResNet-110, indicating a highly controlled error. Table 3.2 presents the ten tightest upper bounds for the selection of the first filter in the specified convolutional layers. As seen, the upper bound on the average absolute error is exceptionally tight (i.e., 1.6e-14), ensuring minimal change in the network when using the ACLI method. From now on, for simplicity, we denote F^l_{SF}

Table 3.2 Upper bound values of the average absolute error, as given by (3.9), for ResNet-50, ResNet-56, and ResNet-110 networks. The results correspond to the ten tightest upper bounds for selecting the first filter

ResNet-50	**Conv 2**	**Conv 4**	**Conv 7**	**Conv 13**	**Conv 17**	**Conv 25**	**Conv 28**	**Conv 31**	**Conv 34**	**Conv 46**
	1.6e-14	8.1e-14	1.9e-13	1.6e-14	2.4e-11	2.7e-13	7.7e-10	1.7e-09	1.7e-09	3.7e-2
ResNet-56	**Conv 1**	**Conv 14**	**Conv 18**	**Conv 24**	**Conv 26**	**Conv 32**	**Conv 40**	**Conv 42**	**Conv 46**	**Conv 52**
	4.8e-12	5.5e-06	1.6e-06	5.1e-09	2.8e-07	1.45e-08	5.1e-05	2.5e-11	4.5e-10	8.4e-07
ResNet-110	**Conv 1**	**Conv 12**	**Conv 32**	**Conv 34**	**Conv 46**	**Conv 52**	**Conv 60**	**Conv 80**	**Conv 92**	**Conv 108**
	7.4e-13	1.0e-11	3.8e-12	9.4e-09	5.3e-10	6.6e-11	1.9e-07	2.7e-11	2.0e-09	4.7e-04

and F^{l+1}_{SC} as $F^l[:,:,:,S]$ and $F^{l+1}[:,:,S,:]$, respectively. In what follows, we present the weakly submodularity of the pruning problem concerning the proposed importance function $\hat{\mathcal{L}}$.

3.3.2 Proposed Pruning as a Weakly Submodular Maximization Problem

In this section, we prove the γ-weakly submodular structure of the importance function $\hat{\mathcal{L}}$. Consider a ground set $V = \{1, 2, \ldots, d\}$ and a set function $\mathcal{F} : 2^V \to \mathbb{R}_+$. We define the marginal gain of including a set $J \subseteq V$ with another set $R \subseteq V$ as $F(J \mid R) = \mathcal{F}(R \cup J) - \mathcal{F}(R)$. This measures the alteration in value upon adding J to R. The size of a set R is denoted as $|R|$.

Definition 3.1. *(El Halabi* et al.*, 2022) Given a set function $\mathcal{F} : 2^V \to \mathbb{R}_+$, a set $U \subseteq V$, and an integer $k \in \mathbb{N}$, we consider that $\mathcal{F}$ is $\gamma_{U,k}$-weakly submodular, with $\gamma_{U,k} > 0$ if*

$$\gamma_{U,k}\mathcal{F}(R \mid L) \leq \sum_{i \in R} \mathcal{F}(i \mid L), \tag{3.11}$$

for every two disjoint sets $L, R \subseteq V$, such that $L \subseteq U$ and $|R| \leq k$. The parameter $\gamma_{U,k}$ is called the submodularity ratio of $\mathcal{F}$.

An efficient method to solve weakly submodular optimization problems and achieve a good approximation to the optimal solution is using a greedy algorithm (Das & Kempe, 2011). The set S of indices from $V = \{1, 2, \ldots, d\}$ is selected using a greedy algorithm, as described in the following steps:

1. *Initialization:* Begin with an empty set of selected indices, $S = \emptyset$.
2. *First index selection:*
   - Evaluate the function $\mathcal{F}$ for each index in V, i.e., $\mathcal{F}(1), \ldots, \mathcal{F}(d)$.
   - Select index $j^* = \text{argmax}\{\mathcal{F}(1), \ldots, \mathcal{F}(d)\}$ and add it to the set S, i.e., $S = \{j^*\}$.
3. *Subsequent index selection:*
   - For each r^{th} index, compute the function values $\mathcal{F}(\{i \cup S\})$ for all $i \in V \backslash S$.
   - Select the index $j^* = \text{argmax}_{i \in V \backslash S} \mathcal{F}(\{i \cup S\})$ and add it to the set S.
4. *Repeat:* Repeat the step 3 until the set S contains k indices, i.e., $|S| = k$.

In each iteration of a greedy algorithm, the most relevant index to the previously selected indices is chosen. Although such a greedy algorithm effectively solves weakly submodular optimization problems and is commonly used by pruning methods (El Halabi *et al.*, 2022), (Luo *et al.*, 2018), its high-complexity iterative procedure is challenging for tiny devices, thereby undermining the purpose of pruning. In this paper, in order to reduce the complexity of filter selection, we utilize a modified version of the greedy algorithm which is not iterative known as the oblivious algorithm. An oblivious algorithm selects a set S^{obv} of k indices with the largest values among the values in the set $\{\mathcal{F}(1), \ldots, \mathcal{F}(d)\}$, obtained in the second step of the greedy algorithm given above, i.e.,

$$\sum_{i \in S^{obv}} \mathcal{F}(i) \geq \sum_{i \in S} \mathcal{F}(i), \tag{3.12}$$

for all $S \subset V$ of size k.

In the following theorem, we prove the submodularity of ACLI importance function. This characteristic allows us to employ an oblivious algorithm for filter selection while achieving a reliable approximation.

Theorem 2. *The importance function $\hat{\mathcal{L}}$ defined in Equation* (3.5) *is a $\gamma_{U,k}$-weakly submodular function.*

Proof: see supplementary material □

Note: In this paper, we assume that for each convolutional layer l ($l = 1, \ldots, \mathbb{L}$) and all indices $i \in \{1, 2, \ldots, H_l\}$, at least one of the norms $\|F^l_{\{i\}F}\|_2^2$ or $\|F^{l+1}_{\{i\}C}\|_2^2$ is nonzero. If both are zero, the filter $F^l_{\{i\}F}$ and its corresponding channels in the $(l+1)^{st}$ layer, $F^{l+1}_{\{i\}C}$, can be eliminated as their presence has no impact on the network.

Note: Let S^* be the optimal index set of k filters for the optimization problem (3.1). For the importance function $\hat{\mathcal{L}}$ defined in (3.5), the following inequalities hold:

$$\gamma_{\emptyset,k} = \min_{R \subset V, |R| \leq k} \frac{\sum_{i \in R} \hat{\mathcal{L}}(i)}{\hat{\mathcal{L}}(R)} \geq 1, \tag{3.13}$$

$$\gamma_{\emptyset,k} \leq \frac{\sum_{i \in S^*} \hat{\mathcal{L}}(i)}{\hat{\mathcal{L}}(S^*)}. \tag{3.14}$$

As demonstrated, our proposed importance function $\hat{\mathcal{L}}$ constitutes a γ-weakly submodular function, as proven in Theorem 2. In the following section, we advocate utilizing the oblivious algorithm to address this weakly submodular maximization for filter selection. Notably, the convergence of this algorithm for weakly submodular functions is assured (Das & Kempe, 2011).

3.4 Oblivious Pruning Algorithm

In this section, we employ the oblivious algorithm to tackle the optimization problem (3.1). Subsequently, we establish in Theorem 3 that the $\gamma_{U,k}$-weak submodularity of the importance function $\hat{\mathcal{L}}$, as defined in (3.5), guarantees the solution obtained by this algorithm, closely approximating the optimal value with an error bounded by $\frac{\gamma_{\emptyset,k}}{k}$, where $\emptyset$ represents the empty set. Algorithm 3.1 presents the oblivious algorithm for selecting the set S^{obv} of filters for pruning from l^{th} convolutional layer, utilizing ACLI proposed method.

Algorithm 3.1 Oblivious algorithm for selecting filters to prune from the l^{th} convolutional layer

Input: The set of filters in the l^{th} and $(l+1)^{st}$ convolutional layers. Pruning rate r;
Output: S^{obv}, the set of selected filters in the l^{th} layer for removal;

$I \leftarrow \{1, 2, \ldots, H_l\}$;
$S^{obv} \leftarrow \emptyset$;
Compute $\hat{\mathcal{L}}(p) = \max(\| F^l_{\{p\}F} \|_2^2, \| F^{l+1}_{\{p\}C} \|_2^2)$ for each $p \in \{1, \ldots, H_l\}$;
while $|S^{obv}| \leq r \cdot H_l$ **do**
    Compute the value of index p^* using Equation (3.15);
    Add p^* to index set S^{obv};
    Remove p^* from I;
end while

3.4.1 Algorithm Procedure

The steps for computing the importance of a single filter in l^{th} convolutional layer is outlined as follows.

- The l_2-norm of each filter $F^l_{\{p\}F}$ and the tensor of its corresponding channels $F^{l+1}_{\{p\}C}$ is computed, for $p = 1, \ldots, H_l$.
- The value $\hat{\mathcal{L}}(p)$ is determined, i.e., the maximum of $\|F^l_{\{p\}F}\|_2^2$ and $\|F^{l+1}_{\{p\}C}\|_2^2$ for $p = 1, \ldots, H_l$.
- The index p^* is chosen for which $\hat{\mathcal{L}}(p^*)$ is minimum in the set $\{\hat{\mathcal{L}}(p) | p = 1, \ldots, H_l\}$.
- The filter $F^l_{\{p^*\}F}$ in the l^{th} convolutional layer is selected for removal.

To generalize this procedure for selecting i^{th} filter $F^l_{\{p_i^*\}F}$ for removal, we use

$$p_i^* = \arg\min_{p \in I} \hat{\mathcal{L}}(p), \tag{3.15}$$

where $i = 1, \ldots, k$, and $I = \{1, \ldots, H_l\} \setminus \{p_1^*, \ldots, p_{i-1}^*\}$, see Figure 3.1. In the following theorem, we establish that Algorithm 3.1 provides an approximation guarantee denoted by $\gamma_{\emptyset,k}$.

Theorem 3. *The set S^{obv} of size k selected by the oblivious algorithm has the following approximation guarantees:*

$$\hat{\mathcal{L}}(S^{obv}) \geq \frac{\gamma_{\emptyset,k}}{k} \hat{\mathcal{L}}(S^*), \tag{3.16}$$

and

$$\hat{\mathcal{L}}(S^{obv}) + \mathcal{R}_{obv} \geq \gamma_{\emptyset,k}\hat{\mathcal{L}}(S^*), \tag{3.17}$$

where S^* is the optimal solution of the problem (3.1)*, and $\mathcal{R}_{obv}$ is defined as $\| F^l_{S_1^{obv}F} \|_2^2 - \| F^{l+1}_{S_1^{obv}C} \|_2^2$ or $\| F^{l+1}_{S_2^{obv}C} \|_2^2 - \| F^l_{S_2^{obv}F} \|_2^2$, depending on whether $\hat{\mathcal{L}}(S^{obv})$ is given by $\| F^l_{S^{obv}F} \|_2^2$ or $\| F^{l+1}_{S^{obv}C} \|_2^2$, respectively, and the sets S_1^{obv} and S_2^{obv} are defined as follows:*

$$S_1^{obv} = \{i \in S \mid\mid \| F^l_{\{i\}F} \|_2^2 \geq \| F^{l+1}_{\{i\}C} \|_2^2\}, \tag{3.18}$$

and

$$S_2^{obv} = \{i \in S \mid\mid \| F^l_{\{i\}F} \|_2^2 \leq \| F^{l+1}_{\{i\}C} \|_2^2\}. \tag{3.19}$$

Proof. see supplementary material. □

The Equation (3.16) illustrates that the set of indices selected by the oblivious algorithm is close to the optimal set S^* by an approximation of $\frac{\gamma_{\emptyset,k}}{k}$. Moreover, Equation (3.17) gives this approximation by two parameters $\gamma_{\emptyset,k}$ and $\mathcal{R}_{obv}$. The closer $\gamma_{\emptyset,k}$ to 1 and $\mathcal{R}_{obv}$ to zero, the closer the selected set to the optimal solution set. As seen, computing $\gamma_{\emptyset,k}$ involves evaluating an NP-hard combinatorial problem over all filter subsets of size at most k, totaling $\sum_{i=0}^{k} \binom{H_l}{i}$ subsets. For instance, the first layer of VGG-16 (with 64 filters) has over 8 million such subsets when $k = 5$. For example, if in the layer l of a CNN, the norm of every filter is equal to the norm of its corresponding channels in the layer $l + 1$, then $\gamma_{\emptyset,k} = 1$, and $\mathcal{R}_{obv} = 0$, therefore the solution of the oblivious algorithm is exactly the optimal solution set, see section 3.7.4 in supplementary material for more examples.

3.4.2 Contribution of $\| F^l_{\{p\}F} \|_2^2$ and $\| F^{l+1}_{\{p\}C} \|_2^2$ in Importance Function $\hat{\mathcal{L}}$

In the oblivious Algorithm 3.1, the importance function $\hat{\mathcal{L}}$ uses the maximum of $\| F^l_{\{p\}F} \|_2^2$ and $\| F^{l+1}_{\{p\}C} \|_2^2$ as the importance value for filter $F^l_{\{p\}F}$. To demonstrate that both norms play a significant role in the filter selection process, we compare the number of filters whose importance values are derived from $\| F^l_{\{p\}F} \|_2^2$ with those derived from $\| F^{l+1}_{\{p\}C} \|_2^2$. Figure 3.2 illustrates

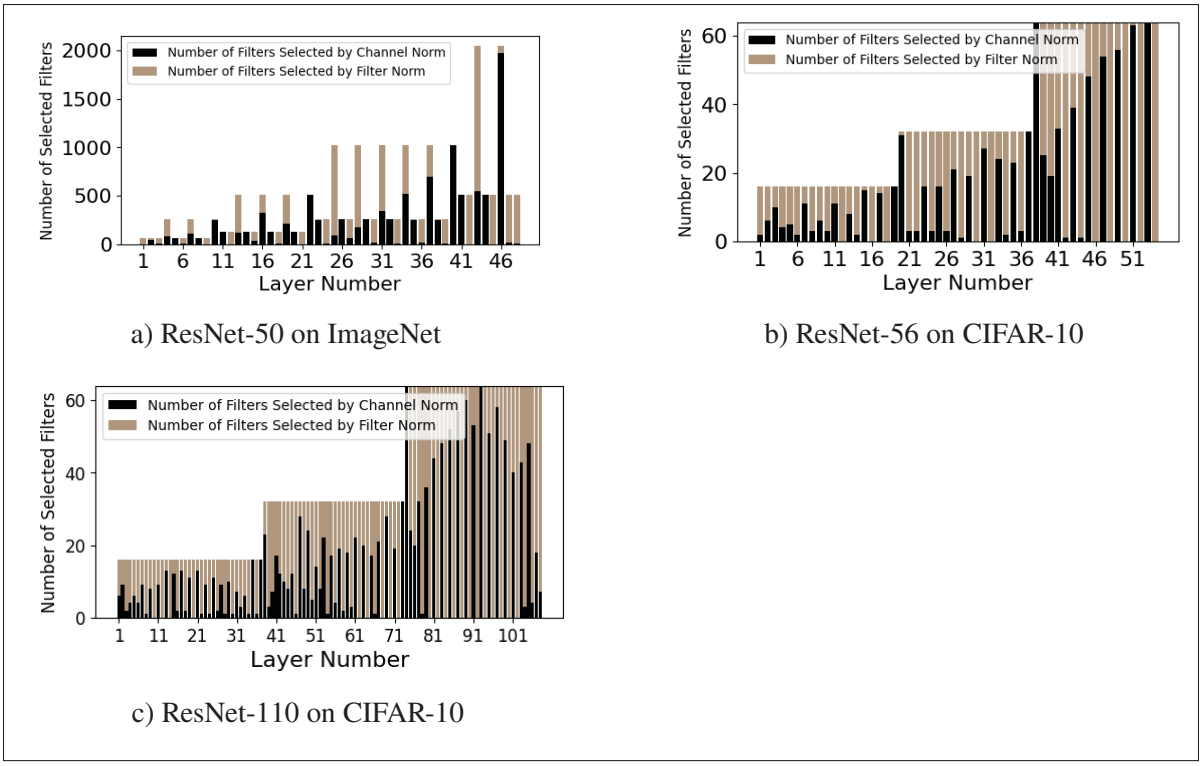

Figure 3.2 The layer-wise contribution of the two types of parameters $F^{l}_{\{i\}T}$ and $F^{l+1}_{\{i\}C}$ in filter selection by $\hat{\mathcal{L}}$ utilizing the oblivious algorithm. The stacked bars compare the number of filters selected by the norm of $F^{l}_{\{i\}T}$ with the number of filters selected by the norm of $F^{l+1}_{\{i\}C}$

this comparison, showing the number of filters whose importance is determined by the norm of $F^{l}_{\{p\}F}$ (i.e., $\| F^{l}_{\{p\}F} \|_2^2 \geq \| F^{l+1}_{\{p\}C} \|_2^2$) versus those determined by the norm of $F^{l+1}_{\{p\}C}$ (i.e., $\| F^{l+1}_{\{p\}C} \|_2^2 \geq \| F^{l}_{\{p\}F} \|_2^2$) across the convolutional layers in ResNet-50 (pre-trained on ImageNet) and ResNet-56 and ResNet-110 (both pre-trained on CIFAR-10).

For example, in the second layer of ResNet-50, 75% of filters attain their importance value based on the norm of the corresponding channels in the third layer,$F^{3}_{\{p\}C}$. According to the results presented by Figure 3.2, both the norms in importance function $\hat{\mathcal{L}}$ contribute effectively in assigning importance value to the filters.

In the following section, we compare the filter selection complexity of different pruning methods, with respect to the number of the multiplications required in the process of the filter selection as the metric for complexity.

Table 3.3 Complexity estimation for selecting k filters in l^{th} convolutional layer

Pruning Method	**Complexity**
ACLI-Oblivious	$\approx H_l(n_l^1 + m_l^1)$
ACLI-Greedy	$\approx \sum_{i=1}^{k}(H_l - i + 1)[n_l^i + m_l^i]$
RED++ (Yvinec *et al.*, 2023)	$\approx \binom{H_l}{2} n_l^1$
HRank (Lin *et al.*, 2020)	$\approx \mathcal{B}\left(\left(\sum_{i=1}^{l-1} C_{i,H_i}\right) + H_l R_l\right)$
F-ThiNet (Tofigh *et al.*, 2022)	$\approx \mathcal{B} H_l N_{l,1,\mathbb{I}}$
ThiNet (Luo *et al.*, 2018)	$\approx \mathcal{B}\sum_{i=1}^{k}(H_l - i + 1)N_{l,i,\mathbb{I}}$
APIB (Guo *et al.*, 2023)	$\approx \left[\left(\mathcal{B}\sum_{i=1}^{l-1} C_{i,H_i}\right) + 2\mathcal{B}^2\mathcal{D}_l + 4\mathcal{B}^3\right]$

3.4.3 Complexity Analysis

Presenting the complexity analysis discussion in Table 3.3, we introduce the following notations. $\hat{r}_l$ is the number of filters kept in l^{th} layer, $\mathcal{B}$ is the batch size, and k_l is the kernel size of filters in l^{th} layer. W_l and L_l are the width and length of the output of the l^{th} layer, respectively. Regarding the baselines presented in Table 3.3, we analyze the complexity of oblivious and greedy algorithms equipped with ACLI importance function $\hat{\mathcal{L}}$, in which the former is our proposed method and the latter serves as one of the baseline methods in our comparison. Moreover, the other baselines, namely RED++ (Yvinec *et al.*, 2023), HRank (Lin *et al.*, 2020), F-ThiNet (Tofigh *et al.*, 2022), ThiNet (Luo *et al.*, 2018), and APIB (Guo *et al.*, 2023), are the most up-to-date efficient baselines regarding the literatures.

The complexities presented in Table 3.3 clearly show that APIB, $\left[\left(\mathcal{B}\sum_{i=1}^{l-1} C_{i,H_i}\right) + 2\mathcal{B}^2\mathcal{D}_l + 4\mathcal{B}^3\right]$, and ThiNet, $\mathcal{B}\sum_{i=1}^{k}(H_l - i + 1)N_{l,i,\mathbb{I}}$, require a significantly greater number of multiplications compared to ACLI method, where $\mathcal{D}_l = 2W_l^2L_l^2 + 2W_{l+1}^2L_{l+1}^2H_{l+1}^2$, $C_{\mu,\nu} = \nu k_\mu^2 H_{\mu-1}L_\mu W_\mu$ $(\mu = 1, ..., \mathbb{L}; \nu = 1, ..., H_\mu)$, $N_{l,\hat{r}_l,\mathbb{I}} = \left(\sum_{i=1}^{l-1} C_{i,H_i}\right) + C_{l,\hat{r}_l} + \hat{r}_l\mathbb{I}k_{l+1}^2$, and $\mathbb{I}$ is the number of the entries in the output of the $(l+1)^{st}$ layer in ThiNet's filter selection procedure. Additionally, HRank, $\mathcal{B}\left(\left(\sum_{i=1}^{l-1} C_{i,H_i}\right) + H_l R_l\right)$, F-ThiNet, $\mathcal{B}H_l N_{l,1,\mathbb{I}}$, and the greedy algorithm equipped

with our importance function $\hat{\mathcal{L}}$, $\sum_{i=1}^{k}(H_l - i + 1)[n_l^i + m_l^i]$, are highly more time-consuming than our proposed oblivious method, $H_l(n_l^1 + m_l^1)$, where $R_l = \sum_{j=1}^{\min(W_l,L_l)-1}(W_l - j)(L_l - j)$, $n_i^j = k_i k_i H_{i-1} j$ and $m_i^j = k_{i+1} k_{i+1} H_{i+1} j$. RED++, $\binom{H_l}{2} n_l^1$, is the closest method to ACLI in terms of complexity of the filter selection, however it is still more complex. It is worth mentioning that the filter selection using the l_2-norm of the filter as the importance criterion (;and Hans Peter Graf, 2017) needs $H_l n_l^1$ multiplication, comparable to ours.

The number of multiplications required by the pruning methods presented in Table 3.3 is calculated as follows:

ACLI - Oblivious: To compute the importance of the filters in the l^{th} layer, both the l_2-norm of the filters in the l^{th} layer and the l_2-norm of their corresponding channels in the filters of the $(l + 1)^{\text{st}}$ layer must be determined, regardless of the number of selected filters, k. In this work, to reduce the computational cost, we use the square of the l_2-norm in place of the l_2-norm, minimizing the number of required multiplications.

The square of the l_2-norm for a filter of size $k_l \times k_l \times H_{l-1}$ involves $k_l^2 H_{l-1}$ multiplications. Consequently, calculating the square of the l_2-norm for all filters in the l^{th} layer and for the corresponding channels in the filters of the $(l + 1)^{\text{st}}$ layer requires $k_l^2 H_{l-1} H_l$ and $k_{l+1}^2 H_l H_{l+1}$ multiplications, respectively. Therefore, the total number of multiplications needed by the oblivious algorithm, combined with our proposed importance function, is $H_l(n_l^1 + m_l^1)$.

ACLI - Greedy: In the greedy algorithm equipped with our proposed importance function, the filters are selected in an iterative manner. In the i^{th} iteration, knowing that already $i - 1$ filters have been selected in the previous iterations as a set S_{i-1}, the next filter is selected by calculating $\hat{\mathcal{L}}(S_{i-1} \cup j)$ for all $j \in V \backslash S_{i-1}$, using the importance function $\hat{\mathcal{L}}$. The number of multiplications required to calculate the importance of $S_{i-1} \cup j$ is $n_l^i + m_l^i = k_l k_l H_{l-1} i + k_{l+1} k_{l+1} i H_{l+1}$. Since in the i^{th} iteration there are $(H_l - i + 1)$ such filters, the total number of the multiplications required by the greedy algorithm for selecting k filters for pruning is $\sum_{i=1}^{k}(H_l - i + 1)[n_l^i + m_l^i]$.

RED++: The pruning process in RED++ consists of three main steps. To estimate the number of multiplications required by RED++ during pruning, we focus on the operations involved in calculating the redundancy of information across filters. In this method, the l_2-norm of the difference between each pair of filters within a convolutional layer is used to quantify their redundancy. Consequently, the number of multiplications needed for this step alone is approximately $\binom{H_l}{2} n_l^1$. It is important to highlight that the actual number of multiplications required by RED++ is higher than this estimate. Nonetheless, even with this lower bound, the complexity of RED++ still remains much greater than that of our proposed oblivious algorithm.

F-ThiNet: In this method, for an input image to the network, the summation of squares for a set of values in the output of the $(l+1)^{\text{st}}$ layer is calculated when only a portion of filters exist in the l^{th} layer. If a set of size $\mathcal{B}$ of images is used in the filter selection process, the computed number of multiplications for one image is multiplied by $\mathcal{B}$. It is worth mentioning that, to obtain the output in the $(l+1)^{\text{st}}$ layer, the input image must pass through all the previous layers, requiring all the convolution operations. Therefore, calculating the number of multiplications, C_{i,H_i}, needed to produce the output of the i^{th} layer for one input image is necessary.

HRank: This method uses the rank of the channels of the output of the i^{th} layer as the importance function. To calculate this rank, the input image passes through all the layers. In order to compute the number of multiplications required by the HRank for filter selection, similar to F-ThiNet and ThiNet, C_{i,H_i} is needed to obtain the output of the i^{th} layer. In the l^{th} layer, for one input image, the calculation of the rank for each of the output channels needs at most

$$\sum_{j=1}^{\min(W_l, L_l)} (W_l - j)(L_l - j),$$

multiplications.

ThiNet: In this method, the computation of the number of multiplications is the same as F-ThiNet, but in the iterative process of the greedy algorithm.

APIB: In this method, to select filters for pruning from the l^{th} layer, we calculate matrices $\bar{L}$, $K^{(k)}$, using the Gaussian kernel function and also matrices Γ, and $\hat{K}^{(k)}$ using matrix multiplication. All these matrices are for a batch of size $\mathcal{B}$. Similarly to the other data-based methods, we also need to pass the input images to the network all the way to the l^{th} layer and all the multiplications must be considered. Note that, to simplify our calculation, we do not even consider the cost of vectorization and solving the optimization problem associated with this method for filter selection.

Table 3.4 Compression results of ResNet-50 on ILSVRC-2012

Model	**Parameters**	**FLOPs**	**Top1 (%)**	**Top-5 (%)**
Baseline (He, Zhang, Ren & Sun, 2016a)	25.5M	4.09B	76.15	92.87
DECORE (Lin *et al.*, 2019)	22.69M	3.54B	76.31	93.02
GAL-0.5 (Lin *et al.*, 2019)	21.20M	2.33B	71.95	90.94
PFP (Liebenwein, Baykal, Lang, Feldman & Rus, 2020)	20.88M	3.65B	75.91	92.81
GAL-0.5 joint (Lin *et al.*, 2019)	19.31M	**1.84B**	71.80	90.82
SSS-32 (Huang & Wang, 2018)	18.60M	2.82B	74.18	91.91
ACLI	**18.22M**	2.86B	**76.52**	**93.18**
PFP (Liebenwein *et al.*, 2020)	17.82M	2.29B	75.21	92.43
HRank (Lin *et al.*, 2020)	16.15M	2.30B	74.98	92.33
He et al. (He, Zhang & Sun, 2017)	-	2.73B	72.30	90.80
SSS-26 (Huang & Wang, 2018)	15.60M	2.33B	71.82	90.70
GDP-0.6 (Lin *et al.*, 2018)	-	1.88B	71.19	90.71
GDP-0.5 (Lin *et al.*, 2018)	-	**1.57B**	69.58	90.14
GAL-1 (Lin *et al.*, 2019)	14.67M	1.58B	69.88	89.75
FilterSketch (Lin *et al.*, 2021)	14.53M	2.23B	74.68	92.17
ACLI	**14.61M**	2.26B	**75.37**	**92.56**
GAL-1 joint (Lin *et al.*, 2019)	10.21M	1.11B	69.31	89.12
ThiNet-50 (Luo *et al.*, 2018)	8.66M	1.10B	68.42	88.30
HRank (Lin *et al.*, 2020)	8.27M	0.98B	69.10	**89.58**
FilterSketch (Lin *et al.*, 2021)	7.18M	0.93B	69.43	89.23
ACLI	**6.67M**	**0.96B**	**69.58**	89.31

3.5 Experiments

3.5.1 Setups

In this section, we assess the performance of our ACLI pruning method under data-free conditions. Specifically, we examine the effectiveness of the one-shot pruning, where a pre-trained model undergoes complete compression in a single step, followed by fine-tuning to restore the network's accuracy. The pruning steps for our proposed method are as follows:

1. *Filter Selection:* The pruning ratio r is uniformly applied across all convolutional layers, except the last one. Using Algorithm 3.1, the filters to be pruned are identified based on their importance scores.
2. *Pruning:* In a single step, the least important filters, identified in the previous phase, are pruned from all convolutional layers, excluding the final layer.
3. *Fine-Tuning:* To restore the network's accuracy post-pruning, the pruned model is fine-tuned. It is worth noting that the ACLI results reported in the following tables, consistent with the publicly available implementations of baseline methods (e.g., NISP (Yu *et al.*, 2018), CHIP (Sui *et al.*, 2021), HRank (Lin *et al.*, 2020)), are obtained using a one-shot pruning strategy rather than an iterative policy. We adopt one-shot pruning in our approach specifically to maintain low computational complexity.

We employ commonly used protocols, namely the number of parameters and required Floating Point Operations (FLOPs), to assess model size and computational demands. Our experiments are carried out on two most popular datasets CIFAR-10 (Krizhevsky, Hinton *et al.*, 2009) and ILSVRC-2012 (Deng *et al.*, 2009). To assess the efficacy and performance of our proposed pruning method, three of the most used networks ResNet-50, ResNet-56, and ResNet-110 are considered, utilizing ResNet-50 on ILSVRC-2012 and ResNet-56 and ResNet-110 on CIFAR-10. In order to evaluate the performance of the pruned networks, the Top-1 and/or Top-5 accuracies are provided. We compare our proposed ACLI pruning method with the baselines including but not limited to HRank (Lin *et al.*, 2020), RED++ (Yvinec *et al.*, 2023), ThiNet (Luo *et al.*, 2018), GAL (Lin *et al.*, 2019), SSS (Huang & Wang, 2018), L1 (;and Hans Peter Graf, 2017), F-ThiNet

(Tofigh *et al.*, 2022), He et al., (He *et al.*, 2017), GDP (Lin *et al.*, 2018), NISP (Yu *et al.*, 2018), DECORE (Alwani *et al.*, 2021), FilterSketch (Lin *et al.*, 2021), and APIB (Guo *et al.*, 2023). We present the results of other methods based on their respective literature, marking them as '-' if not reported. The best pruning results are highlighted in bold.

Table 3.5 Compression results of ResNet-56 on CIFAR-10

Model	**Parameters**	**FLOPs**	**Top1 (%)**
Baseline (He *et al.*, 2016a)	0.85M	125.49M(0.0%)	93.26
GAL-0.6 (Lin *et al.*, 2019)	0.75M (11.8%)	78.30M (37.6%)	92.98
L1 (;and Hans Peter Graf, 2017)	0.73M (14.1%)	90.90M (27.6%)	93.06
ACLI	**0.71M (16.8%)**	104.15M (17%)	**94.09**
HRank (Lin *et al.*, 2020)	0.71M(16.8%)	88.72M(29.3%)	93.52
FilterSketch (Lin *et al.*, 2021)	0.68M(20.6%)	88.05M(30.43%)	93.65
DECORE (Lin *et al.*, 2019)	0.64M (24.2%)	92.48(26.3%)	93.34
ACLI	**0.59M (30.43%)**	86.85(30.78%)	**93.84**
SOKS (Liu, Zhang & Lv, 2022)	0.51M (40%)	81.40M (36%)	93.22
FilterSketch (Lin *et al.*, 2021)	0.50M (41.2%)	73.36M (41.5%)	93.19
NISP (Yu *et al.*, 2018)	0.49M (42.4%)	81.00M (35.5%)	93.01
HRank (Lin *et al.*, 2020)	0.49M (42.4%)	62.72M (50.0%)	93.17
HTP-URC (Qian *et al.*, 2023)	0.47M (44.72%)	58.65M (46.58%)	93.55
DECORE (Lin *et al.*, 2019)	0.43M (49%)	62.93M (49.9%)	93.26
ACLI	**0.42M (49.97%)**	62.12M (50.5%)	**93.54**
GAL-0.8 (Lin *et al.*, 2019)	0.29M (65.9%)	49.99M (60.2%)	90.36
HRank (Lin *et al.*, 2020)	0.27M (68.1%)	32.52M (71.69%)	90.72
QSFM (Wang *et al.*, 2022)	0.25M (71%)	50.62M (60%)	91.88
CHIP (Sui *et al.*, 2021)	0.24M (71.8%)	34.79M (72.3%)	92.05
FilterSketch (Lin *et al.*, 2021)	0.24M (71.8%)	32.47M (74.4%)	91.2
ACLI	**0.22M (74.12%)**	32.18M (74.36%)	**92.56**
APIB (Guo *et al.*, 2023)	0.14M (83%)	23.85M (81%)	**91.53**
ACLI	**0.136M (84%)**	20.08M (84%)	91.39

Configuration: For implementing these evaluations, the code is run on Python 3.9.12 using PyTorch framework 1.11.0. We also use cuDNN 8200 to handle fine-tuning of these deep networks. The IDE used in all the implemented codes is PyCharm. To implement our method for pruning ResNet-50 on ImageNet, we use the pre-trained network available by PyTorch. The learning rate in this experiment is 0.001 and is adjusted every 30 epochs. To implement our method for pruning ResNet-56 and ResNet-110 on CIFAR-10, we utilize the pre-trained network

Table 3.6 Pruning results for ResNet-110 on CIFAR-10

Model	**Parameters**	**FLOPs**	**Top1 (%)**
Baseline (He *et al.*, 2016a)	1.7M(0.0%)	252.89M(0.0%)	93.50
L1 (;and Hans Peter Graf, 2017)	1.16M (32.6%)	155M (38.7 %)	93.3
DECORE (Alwani *et al.*, 2021)	1.11M (36%)	163.30M (35%)	93.88
HRank (Lin *et al.*, 2020)	1.04M(39.4%)	148.70M(41.2%)	94.23
ACLI	0.97M (43.3%)	142.804M (43.3%)	94.2
GAL-0.5 (Lin *et al.*, 2019)	0.95M(44.8%)	130.20M(48.5%)	92.55
HRank (Lin *et al.*, 2020)	0.70M(59.2%)	105.70M(58.2%)	93.36
ACLI	**0.69M**(59.7%)	101.09M(60.0%)	**93.91**
HRank (Lin *et al.*, 2020)	0.53M(68.7%)	79.30M(68.6%)	92.65
ACLI	**0.53M** (69.2%)	77.284M (69.4%)	**93.28**

provided by (He *et al.*, 2019b). The learning rate for this experiment is set to 0.001, and it is adjusted between 0.0001 and 0.00001. A batch size of 64 is used for the fine-tuning.

3.5.2 Experimental Results

We evaluate the compression of ACLI method on ResNet-50, ResNet-56, and ResNet-110 with results presented in Tables 3.4, 3.5, and 3.6. In each table, the baseline represents the number of parameters, FLOPs, and the accuracy of the pre-trained model without pruning.

Table 3.7 Resource Efficiency of ResNet-50 on ImageNet

Model	**Pruning Ratio (r)**	**RE**
ACLI	0.2	4.4e-07
	0.32	4.3e-07
	0.72	4.0e-07
RED++ (Yvinec *et al.*, 2023)	0.33	2.1e-09
HRank (Lin *et al.*, 2020)	0.25	1.0e-12
	0.35	9.9e-13
	0.6	9.5e-13
IPBM (Guo *et al.*, 2023)	0.39	1.1e-16
	0.48	1.1e-16
ThiNet	0.3	6.6e-18
	0.5	4.3e-18
	0.7	3.4e-18

Table 3.8 Resource Efficiency of ResNet-56 on CIFAR-10

Model	Pruning Ratio (r)	RE
ACLI	0.09	5.7e-05
	0.17	5.7e-05
	0.25	5.6e-05
	0.5	5.5e-05
	0.6	5.5e-05
RED++ (Yvinec *et al.*, 2023)	0.68	4.1e-06
HRank (Lin *et al.*, 2020)	0.09	1.1e-10
	0.25	1.1e-10
	0.48	1.2e-11
IPBM (Guo *et al.*, 2023)	0.33	5.6e-14
	0.43	5.6e-14
	0.57	5.5e-14

Table 3.9 Resource Efficiency of ResNet-110 on CIFAR-10

Model	Pruning Ratio (r)	RE
ACLI	0.25	2.8e-05
	0.37	2.8e-05
	0.45	2.7e-05
RED++ (Yvinec *et al.*, 2023)	0.72	2.0e-06
HRank (Lin *et al.*, 2020)	0.23	2.8e-11
	0.37	2.7e-11
	0.44	2.7e-11
IPBM (Guo *et al.*, 2023)	0.41	2.8e-14
	0.58	2.8e-14

ResNet-50 on ImageNet: As shown in Table 3.4, our ACLI approach achieves an outstanding Top-1 accuracy of 76.52%, surpassing the baseline model (76.52% vs. 76.15%) with similar FLOPs and parameter reduction as SSS-32. Notably, ACLI yields a significantly greater reduction in parameter count and nearly identical reduction in FLOPs compared to FilterSketch, while achieving superior accuracy (75.37% vs. 74.68%). Moreover, even with extensive pruning, ACLI remains better than top-1 accuracy, including GAL-1 joint, ThiNet-50, HRank, and FilterSketch, while achieving greater reductions in parameter count and FLOPs (69.58% vs. 69.31% for

GAL-1 joint, 68.42% for ThiNet-50, 69.1% for HRank, and 69.43% for FilterSketch). Since no numerical results are reported by RED++ (Yvinec *et al.*, 2023) for ResNet-50 on ImageNet, we compare ACLI with RED++ using their metric: the percentage of preserved accuracy relative to the percentage reduction in parameters. As shown in Table 3.4, ACLI preserves almost 100% of the accuracy while achieving a 57% reduction in parameters, outperforming all the results presented in (Yvinec *et al.*, 2023). Moreover, for 28.5% reduction in the parameters, the percentage of preserved accuracy by ACLI is around 100.5%.

ResNet-56 on CIFAR-10: Consistent with our findings on ResNet-50, ACLI demonstrates an accuracy improvement over the baseline model (94.09% vs. 93.26%) while achieving approximately 16.8% and 17% reduction in FLOPs and parameters, respectively. Furthermore, compared to HRank, FilterSketch, and DECORE, which offer marginally better complexity reduction, ACLI method consistently delivers superior accuracy (93.84% vs. 93.52% for HRank, 93.65% for FilterSketch, and 93.34% for DECORE). Notably, for extensive pruning, ACLI reduces model complexity compared to APIB (84% vs. 83% in terms of parameters, and 84% vs. 81% in terms of FLOPs) with similar accuracy of the pruned network.

ResNet-110 on CIFAR-10: ACLI consistently outperforms other approaches across all evaluated scenarios, excelling in both performance and complexity reduction. Our method achieves a 59.7% reduction in parameters while also delivering a 0.41% accuracy increase compared to the leading pruning methods, HRank (Lin *et al.*, 2020) and DECORE (Alwani *et al.*, 2021). When compared to other methods like L1 (;and Hans Peter Graf, 2017) and GAL (Lin *et al.*, 2019), ACLI offers higher accuracy with lower computational cost and fewer parameters.

Other CNNs: To evaluate the performance of ACLI on resource-efficient models, we applied our method to several lightweight networks. Table 3.10 presents the corresponding results.

3.5.3 Resource Efficiency

While maintaining the accuracy of the pruned network is crucial, a pruning method needs also be practical and suitable for deployment on devices. In other words, it must be low-complexity and

Network	Dataset	Pruning Ratio	ACC (%)
Tiny-YOLO	CIFAR-10	25%	0.42% ↓
Mobilenet-v2	CIFAR-10	40%	1.18% ↓
VGG-11	MNIST	50%	0.12% ↑
AlexNet	CIFAR-10	50%	5.00% ↑

Table 3.10 Performance evaluation on lightweight networks

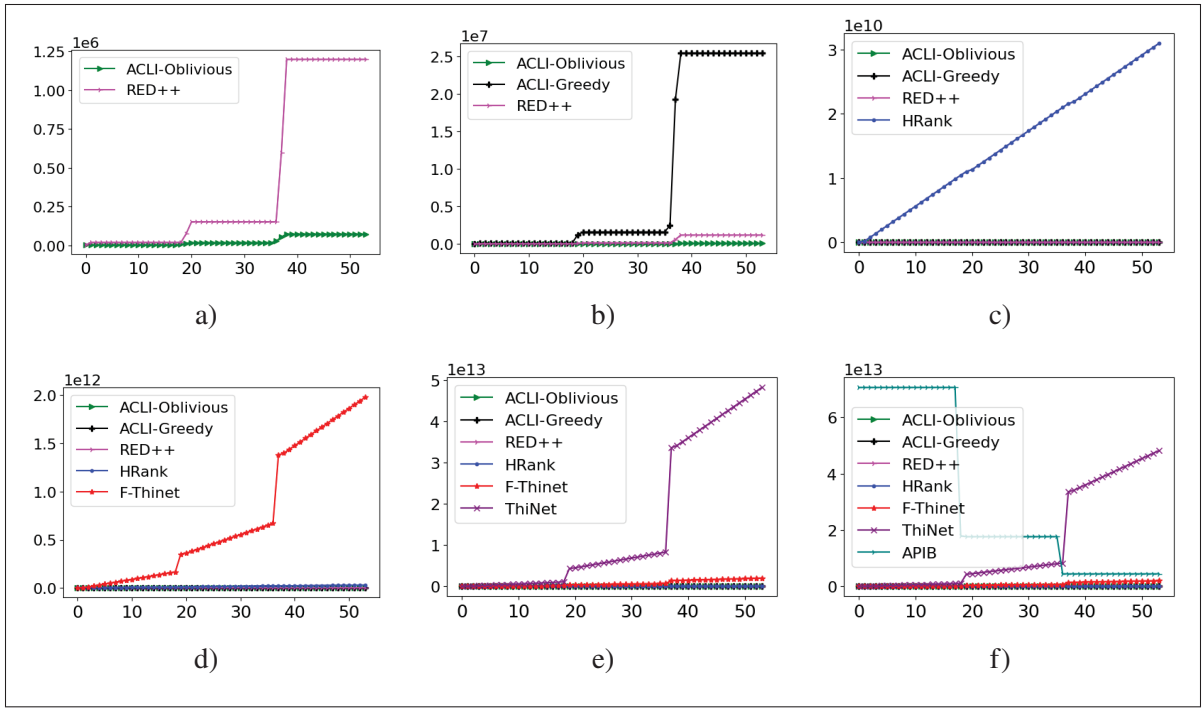

Figure 3.3 Complexity comparison of different pruning methods for ResNet-56 on CIFAR-10. The X-axis represents the layer number, and the Y-axis represents the number of multiplications required in the filter selection

fast, requiring minimal device resources during the pruning process. To address this, we propose a metric for evaluating the resource efficiency (RE) of a pruning method, which considers both accuracy and the computational complexity of the pruning process. This metric is defined as:

$$RE = \left(\frac{\text{ACC}}{\text{MAC}}\right), \tag{3.20}$$

Table 3.11 Ratio comparison of the number of multiplications required by the pruning methods

	ACLI (Oblivious)	**ACLI (Greedy)**	**F-ThiNet**	**ThiNet**	**HRank**	**APIB**	**RED++**
ACLI(Obl)	1	3.62e-03	4.25e-08	2.04e-09	2.01e-06	9.94e-10	7.28e-02
ACLI(Gr)	2.76e+02	1	1.17e-05	5.62e-07	5.56e-04	2.75e-07	2.01e+01
F-ThiNet	2.35e+07	8.53e+04	1	4.79e-02	4.74e+01	2.34e-02	1.71e+06
ThiNet	4.91e+08	1.78e+06	2.09e+01	1	9.88e+02	4.88e-01	3.58e+07
HRank	4.97e+05	1.80e+03	2.11e-02	1.01e-03	1	4.94e-04	3.62e+04
APIB	1.01e+09	3.64e+06	4.27e+01	2.05e+00	2.02e+03	1	7.32e+07
RED++	1.37e+01	4.97e-02	5.83e-07	2.80e-08	2.76e-05	1.37e-08	1

where ACC represents the accuracy of the pruned network and MAC denotes the number of multiply-accumulate operations required in the pruning process. This metric provides a balanced evaluation of pruning methods, highlighting the trade-off between preserving accuracy and reducing computational costs, thereby offering a comprehensive measure of a method's overall efficiency. Tables 3.7, 3.8, and 3.9 present the RE for ACLI and the main baselines, HRank (Lin *et al.*, 2020), ThiNet (Luo *et al.*, 2018), and IPBM (Guo *et al.*, 2023), applied to ResNet-50 on ImageNet, ResNet-56 on CIFAR-10, and ResNet-110 on CIFAR-10. In this experiment, the accuracy of the pruning methods is sourced from the literature. Since the pruning ratios corresponding to each accuracy are not explicitly provided in the literature, we approximate them using the number of parameters reduced for each accuracy. Due to the absence of numerical results for ResNet-56 and ResNet-110 for ThiNet (Luo *et al.*, 2018), it is only included in Table 3.7 for ResNet-50. Similarly, RED++ offers graphical results for these networks; therefore, we estimate the accuracy preservation for RED++ to be around 95.5%. The percentage of parameters removed is also approximated from the graphs in the literature. When calculating the MAC for RED++, we only account for the computations involved in the redundancy evaluation step during the similarity-based clustering. As a result, the actual MAC of RED++ is likely higher than the values presented here.

The results presented in Tables 3.7, 3.8, and 3.9 demonstrate that the resource efficiency of each pruning method remains relatively consistent across different pruning ratios. For instance, the RE of the ACLI method on ResNet-50 is approximately 4e-7, regardless of the pruning ratio,

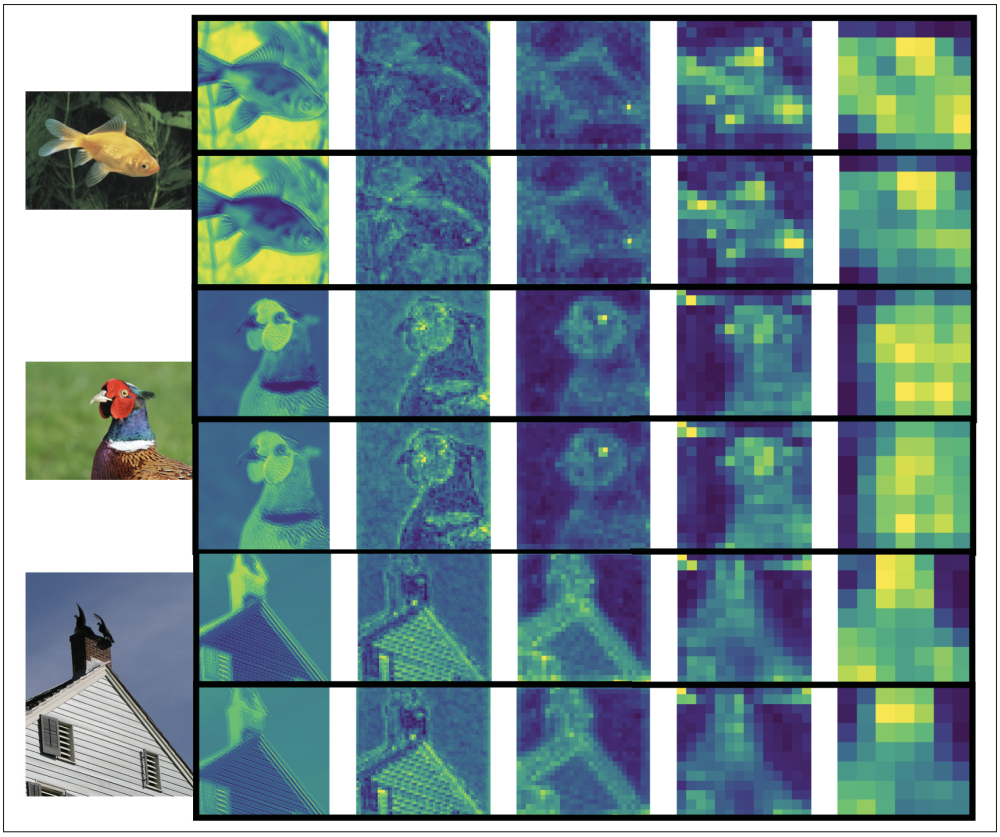

Figure 3.4 Style transfer results using ResNet-50 on three randomly selected images from the ImageNet dataset. For each image, the top row shows the output results from the first five convolutional layers of the original ResNet-50, while the bottom row displays the corresponding results from the pruned network using ACLI with pruning ratio $r = 0.2$

within the range of 0.2 to 0.72. This trend is similarly observed across other networks and pruning methods, suggesting that RE serves as a reliable metric for evaluating the computational cost of a pruning method relative to the accuracy achieved.

Based on the RE values presented in Tables 3.7, 3.8, and 3.9, the pruning method closest in performance to ACLI is RED++, which consistently achieves approximately one-tenth of ACLI's resource efficiency across all tested networks. In comparison, the resource efficiency of APIB is substantially lower, with ACLI demonstrating an RE approximately e+09 times higher. Similarly, HRank exhibits a much lower RE, with ACLI outperforming it by a factor of e+05. These results

highlight the significant advantage of the proposed ACLI method in terms of resource efficiency, underscoring its superior balance between network accuracy preservation and computational complexity.

3.5.4 Complexity Comparison

In this section, we compare the complexity of various pruning methods for filter selection in convolutional layers of ResNet-56 on CIFAR-10, presenting the results in Figure 3.3. The number of multiplications in the process of the filter selection is used as the complexity metric. In all the results presented in this section, a 50% pruning ratio and a batch size of 256 is considered. For the complexity of ThiNet and F-ThiNet methods, the parameter $\mathbb{I}$, the number of entries in the output of $(l+1)^{st}$ layer, is 10. Moreover, we compare the ratio of complexity for filter selection in the convolutional layers of ResNet-56 on CIFAR-10 across different pruning methods, Table 3.11. The value in the i^{th} row and j^{th} column of the table is calculated as $\frac{N_R(i)}{N_C(j)}$, where $N_R(i)$ and $N_C(j)$ represent the total number of multiplications required for filter selection by the method in the i^{th} row and j^{th} column, respectively.

Overall, Figure 3.3 shows a significant difference in complexity between our oblivious algorithm and other pruning methods. Figure 3.3a compares the computational complexity of ACLI-oblivious and RED++. The ACLI method requires a total of 1.7e+06 multiplications, whereas RED++ demands 2.3e+07, making RED++ at least 13.7 times more computationally complex than ACLI-oblivious. Figure 3.3b compares the complexity of the oblivious and greedy algorithms equipped with our proposed importance function and RED++. As shown, the layer-wise complexity of the greedy algorithm is significantly higher than the oblivious algorithm, resulting 2.76e+02 times more required multiplications in total. Adding the results for the greedy algorithm in Figure 3.3b nearly flattens the graph for the oblivious algorithm, as the greedy algorithm's curve exhibits significantly higher values. In Figures 3.3c, 3.3d, and 3.3e, an increase in complexity is observed with the inclusion of results for HRank, F-ThiNet, and ThiNet, respectively. According to Figure 3.3f, APIB demands more multiplications than the

other methods, except for ThiNet in the deeper layers, requiring a total of 1.01e+09 times more multiplications than the ACLI-oblivious.

As seen in the first row of Table 3.11, the total number of multiplications needed by the oblivious algorithm is 3.62e-3 times that required by the greedy algorithm. The ratio is 4.25e-08, 2.04e-09, 2.01e-06, and 9.094e-10 for F-ThiNet, ThiNet, HRank, and APIB, respectively. Similarly, the number in the sixth row and first column of Table 3.11 represents that the number of multiplications required by APIB is almost e+09 times that required by the oblivious algorithm, highlighting the vast disparity in the running times of these two algorithms. The ratios of the number of multiplications required by greedy, F-ThiNet, ThiNet, HRank to that of oblivious are 2.76e+02, 2.35e+07, 4.91e+08, and 4.97e+05, respectively.

3.5.5 Visualizing Feature Preservation

We qualitatively evaluate the feature retention of the proposed ACLI pruning method on ResNet-50 using the ImageNet dataset. We randomly selected three images from ImageNet and compared the features extracted by the first five convolutional layers of the pruned network (with $r = 0.2$) to those of the original network. For each image, the top row displays the features of the original network, while the bottom row shows the features of the pruned network. As observed in Figure 3.4, there is no significant difference between the features extracted in these two scenarios.

3.6 Conclusion

In this paper, we presented a low-complex pruning technique based on defining a novel and efficient importance function, leveraging a proven upper bound inequality on the average absolute error of the output of the layer succeeding the pruned layer. Presenting the pruning problem as a weakly submodular optimization, we proposed using an oblivion algorithm, which offers reduced complexity compared to other widely-used pruning methods, requiring only a fraction of the

computation. Despite its simplified algorithmic framework, our pruning technique outperformed state-of-the-art methods in empirical performance.

3.7 Supplementary Materials

3.7.1 Proof of Theorem 1

In this paper, we use the regular definition of convolution operation in a convolutional layer. The $[m, n, s]$ entry in the output of l^{th} convolutional layer is obtained by convolving the s^{th} filter, F^l_{sF}, by part of the input, $\bigoplus_l \overleftrightarrow{(m, n, s)}$. The convolution operation is the inner product of F^l_{sF} and $\bigoplus_l \overleftrightarrow{(m, n, s)}$. In order to proof the theorem, we calculate the absolute error in the output of the $(l + 1)^{st}$ layer. The $[m, n, s]$ entry in the output of the $(l + 1)^{st}$ layer is as follows

$$X_{l+1}[m, n, s] = \left\langle F^{l+1}_{\{s\}F}, ReLu\left(\bigoplus_l \overleftrightarrow{(m, n, s)}\right)\right\rangle = \sum_{p=1}^{H_l}\sum_{i,j=1}^{k_{l+1}} f^{l+1}_{(i,j,p,s)} ReLu(\mathcal{A}^{m,n,s}_{i,j,p}), \tag{3.21}$$

where

$$\mathcal{A}^{m,n,s}_{i,j,p} = \left(\bigoplus_l \overleftrightarrow{(m, n, s)}\right)[i, j, p] = \left\langle F^l_{pF}, Relu\left(\bigoplus_{l-1}\left(\bigoplus_l \overleftrightarrow{(m, n, s)}[i, j, p]\right)\right)\right\rangle. \tag{3.22}$$

Similarly, $\hat{X}^{p_r}_{l+1}[m, n, s]$, the $[m, n, s]$ entry of the X_{l+1} when the p^{th}_r filter from the l^{th} layer is pruned, is as follows

$$\hat{X}^{p_r}_{l+1}[m, n, s] = \sum_{p=1}^{p_r-1}\sum_{i,j=1}^{k_{l+1}} f^{l+1}_{(i,j,p,s)} ReLu(\mathcal{A}^{m,n,s}_{i,j,p}) + \sum_{p=1}^{p_r+1}\sum_{i,j=1}^{k_{l+1}} f^{l+1}_{(i,j,p,s)} ReLu(\mathcal{A}^{m,n,s}_{i,j,p}) \tag{3.23}$$

Then the absolute error in the the $[m, n, s]$ entry of X_{l+1} before and after pruning is calculated as follows

$$|X_{l+1}[m,n,s] - \hat{X}^{p_r}_{l+1}[m,n,s]|^2 = \sum_{i,j=1}^{k_{l+1}} f^{l+1}_{(i,j,p_r,s)} ReLu(\mathcal{A}^{m,n,s}_{i,j,p})$$

$$\leq \left(\sum_{i,j=1}^{k_{l+1}} \left(f^{l+1}_{(i,j,p_r,s)}\right)^2\right)\left(\sum_{i,j=1}^{k_{l+1}} (\mathcal{A}^{m,n,s}_{i,j,p})^2\right) \tag{3.24}$$

$$\leq \| F^{l+1}_{(:,:,p_r,s)} \|_2^2 \| F^{l}_{(:,:,:,p_r)} \|_2^2 \, \alpha^{l-1}_{(m,n)}, \tag{3.25}$$

where

$$\alpha^{l-1}_{(m,n)} = \sum_{i=1}^{k_{l+1}} \sum_{j=1}^{k_{l+1}} \left(\| \, Relu\left(\bigoplus_{l-1} \left(\bigoplus_{l} \overleftrightarrow{(m,n)}[i,j,p_r] \right) \right) \|_2^2 \right). \tag{3.26}$$

Equations (3.24) and (3.25) can be hold from Cauchy-Schwarz inequality and lemma 1 proven in (Srinivas & Babu, 2015).

Therefore, the inequality in Theorem 1 holds by summing over the values of m, n, and s as follows

$$\| X_{l+1} - \hat{X}^{p_r}_{l+1} \|_2^2 = \sum_{s=1}^{H_{l+1}} \sum_{m=1}^{W_{l+1}} \sum_{n=1}^{L_{l+1}} |X_{l+1}[m,n,s] - \hat{X}^{p_r}_{l+1}[m,n,s]|^2 \tag{3.27}$$

$$\leq \sum_{s=1}^{H_{l+1}} \sum_{m=1}^{W_{l+1}} \sum_{n=1}^{L_{l+1}} \| F^{l+1}_{(:,:,p_r,s)} \|_2^2 \| F^{l}_{(:,:,:,p_r)} \|_2^2 \, \alpha^{l-1}_{(m,n)} \tag{3.28}$$

$$= \| F^{l+1}_{(:,:,p_r,:)} \|_2^2 \| F^{l}_{(:,:,:,p_r)} \|_2^2 \, \Lambda_l, \tag{3.29}$$

where

$$\Lambda_l = \sum_{m=1}^{W_{l+1}} \sum_{n=1}^{L_{l+1}} \alpha^{l-1}_{(m,n)}. \tag{3.30}$$

3.7.2 Proof of Theorem 2

Proof. Given set $V = \{1, 2, \ldots, H_l\}$, the set of indices of the filters in l^{th} convolutional layer, for every $U \subset V$, $L \subset U$, and $i \notin L$

$$\hat{\mathcal{L}}(i|L) = \max\{\| F^l_{(L\cup\{i\})F} \|_2^2, \| F^{l+1}_{(L\cup\{i\})C} \|_2^2\} \tag{3.31}$$

$$- \max\{\| F^l_{LF} \|_2^2, \| F^{l+1}_{LC} \|_2^2\}$$

$$= \max\{\| F^l_{LF} \|_2^2 + \| F^l_{\{i\}F} \|_2^2 \tag{3.32}$$

$$, \| F^{l+1}_{LC} \|_2^2 + \| F^{l+1}_{\{i\}C} \|_2^2\}$$

$$- \max\{\| F^l_{LF} \|_2^2, \| F^{l+1}_{LC} \|_2^2\} > 0. \tag{3.33}$$

Now, let's assume that $R \subset V$ is a set that is disjoint from the set U such that $|R| \leq k$. Therefore, $\sum_{i\in R} \hat{\mathcal{L}}(i|L) > 0$. Setting $\gamma_{U,k}$ as follows

$$\gamma_{U,k} = \min_{L\subset U} \min_{R\subset V, |R|\leq k} \frac{\sum_{i\in R} \hat{\mathcal{L}}(i|L)}{\hat{\mathcal{L}}(R|L)}, \tag{3.34}$$

the inequality condition in Equation 11 is satisfied. Because U has finite subsets, and for every set $L \subset U$ there is a finite number of such sets R, so, the feasible set in Equation (3.34) is finite. As a result, a positive number $\gamma_{U,k}$ exists. Since for every set $U \subset \{1, 2, \ldots, H_l\}$ and every integer number $k \in \mathbb{N}^+$, the submodularity ratio $\gamma_{U,k}$ exists, the set function $\hat{\mathcal{L}}$ is a $\gamma_{U,k}$-weakly submodular.

3.7.3 Proof of Theorem 3

Proof. Let S^{obv} denote the set selected by the oblivious algorithm, and let S^* represent the optimal index set of k filters. By definition of the oblivious algorithm ($\sum_{i\in S^{obv}} \hat{\mathcal{L}}(i) \geq \sum_{i\in S^*} \hat{\mathcal{L}}(i)$), using Equations (3.6) and (3.14), the following inequalities are hold

$$\hat{\mathcal{L}}(S^{obv}) \geq \frac{\sum_{i\in S^{obv}} \hat{\mathcal{L}}(i)}{k} \geq \frac{\sum_{i\in S^*} \hat{\mathcal{L}}(i)}{k} \geq \frac{\gamma_{\emptyset,k}}{k} \hat{\mathcal{L}}(S^*). \tag{3.35}$$

which proves Equation (3.16). To prove Equation (3.17), without loss of generality, we assume $\hat{\mathcal{L}}(S^{obv}) = \| F^l_{S^{obv}F} \|_2^2$, therefore

$$\hat{\mathcal{L}}(S^{obv}) = \| F^l_{S^{obv}F} \|_2^2 \tag{3.36}$$

$$= \| F^l_{S_1^{obv}F} \|_2^2 + \| F^l_{S_2^{obv}F} \|_2^2 \tag{3.37}$$

$$\leq \| F^l_{S_1^{obv}F} \|_2^2 + \| F^{l+1}_{S_2^{obv}C} \|_2^2 . \tag{3.38}$$

Since $\hat{\mathcal{L}}(S^{obv}) = \| F^l_{S^{obv}F} \|_2^2$, consequently

$$\begin{aligned}
&\| F^l_{S_1^{obv}F} \|_2^2 + \| F^l_{S_2^{obv}F} \|_2^2 \geq \| F^{l+1}_{S_1^{obv}C} \|_2^2 + \| F^{l+1}_{S_2^{obv}C} \|_2^2 \\
\Rightarrow &2 \| F^l_{S_1^{obv}F} \|_2^2 + \| F^l_{S_2^{obv}F} \|_2^2 - \| F^{l+1}_{S_1^{obv}C} \|_2^2 \\
&\geq \| F^l_{S_1^{obv}F} \|_2^2 + \| F^{l+1}_{S_2^{obv}C} \|_2^2 \qquad (3.39) \\
\Rightarrow &\hat{\mathcal{L}}(S^{obv}) + (\| F^l_{S_1^{obv}F} \|_2^2 - \| F^{l+1}_{S_1^{obv}C} \|_2^2) \\
&\geq \sum_{i \in S^{obv}} \hat{\mathcal{L}}(i) \geq \sum_{i \in S^*} \hat{\mathcal{L}}(i) \geq \gamma_{\emptyset,k} \hat{\mathcal{L}}(S^*), \qquad (3.40)
\end{aligned}$$

now by defining $\mathcal{R}_{obv} = \| F^l_{S_1^{obv}F} \|_2^2 - \| F^{l+1}_{S_1^{obv}C} \|_2^2$, Equation (3.17) is proven. Similarly by assuming $\hat{\mathcal{L}}(S^{obv}) = \| F^{l+1}_{S^{obv}C} \|_2^2$ the inequality hold for $\mathcal{R}_{obv} = \| F^{l+1}_{S_2^{obv}C} \|_2^2 - \| F^l_{S_2^{obv}F} \|_2^2$.

3.7.4 Submodularity of $\| . \|_2^2$

In this section, we explore the submodularity of $\| . \|_2^2$ as a set function on the set of filters in a convolutional layer. For the set of filters in the l^{th} layer of a CNN, the function $\| . \|_2^2$ can be regarded as a set function. Given a subset S of $\{1, 2, \ldots, H_l\}$, the index set of the filters in the l^{th} layer, it returns $\| F^l[:, :, :, S] \|_2^2$. The following lemma can be easily proven for this set function.

Lemma 1. *For any subset $S \subset \{1, 2, \ldots, H_l\}$ and any index $\alpha \notin S$ we have:*

$$\| F^l[:, :, :, S \cup \{\alpha\}] \|_2^2 - \| F^l[:, :, :, S] \|_2^2 = \| F^l[:, :, :, \alpha] \|_2^2, \tag{3.41}$$

and

$$\| F^{l+1}[:,:,S \cup \{\alpha\},:] \|_2^2 - \| F^{l+1}[:,:,S,:] \|_2^2 = \| F^{l+1}[:,:,\alpha,:] \|_2^2 . \tag{3.42}$$

The following theorem shows that $\| . \|_2^2$ is a submodular set function.

Theorem 4. *The submodularity ratio of the set function $\| . \|_2^2$ is 1. The set of size K selected by the oblivious algorithm equipped with the $\| . \|_2^2$, as the importance function, is the optimal solution.*

Proof: Let's assume $U \subset V$. For any $L \subset U$ and $S \subset V$ where $|S| \leq k$

$$\gamma_{U,k} = \min_{L \subset U} \min_{S \subset V, |S| \leq k} \left(\frac{\sum_{i \in S} \| i|L \|_2^2}{\| S|L \|_2^2} \right) \tag{3.43}$$

$$= \min_{L \subset U} \min_{S \subset V, |S| \leq k} \left(\frac{\sum_{i \in S} \| F^l_{(L \cup \{i\})F} \|_2^2 - \| F^l_{LF} \|_2^2}{\| F^l_{(L \cup \{S\})F} \|^2 - \| F^l_{LF} \|^2} \right) \tag{3.44}$$

$$= \min_{L \subset U} \min_{S \subset V, |S| \leq k} \left(\frac{\sum_{i \in S} \| F^l_{\{i\}F} \|_2^2}{\| F^l_{SF} \|_2^2} \right) = 1. \tag{3.45}$$

This theorem shows that when the filter selection criteria is only the l_2-norm of the filters, the $\gamma_{U,k} = 1$. It means that the $\| . \|_2^2$ is a submodular set function . On the other hand, since

$$\sum_{i \in S} \| F^l_{\{i\}F} \|_2^2 = \| F^l_{SF} \|_2^2, \tag{3.46}$$

a set S of filters has the maximum l_2-norm when the filters in S individually have the greatest norms among the filters in the l^{th} layer. Therefore, the oblivious algorithm equipped with $\| \ldots \|_2^2$ provides the optimal solution to the maximization problem (1) in the paper.

Remark 1. *In a convolutional layer, if $\| F^l_{\{i\}F} \|_2^2 \geq \| F^{l+1}_{\{i\}C} \|_2^2$ for all $i = 1, \ldots, H_l$, then the importance function $\hat{\mathcal{L}}$ is submodular, and $S^{obv} = S^*$.*

Remark 2. *In a convolutional layer, if* $\| F^{l}_{\{i\}F} \|_2^2 \leq \| F^{l+1}_{\{i\}C} \|_2^2$ *for all* $i = 1, \ldots, H_l$*, then the importance function* $\hat{\mathcal{L}}$ *is submodular, and* $S^{obv} = S^{*}$.

3.7.5 Experimental setup

Our code uses Pytorch 1.11.0. Our implementation is adapted from the code provided in (He, Liu, Wang, Hu & Yang, 2019a), for both ResNet-50 on ImageNet and ResNet-56 on CIFAR-10. The used pre-trained ResNet-50 on ImageNet is the pretarined model provided by PyTorch and the for ResNet-56 on CIFAR-10 we used the pre-trained model provided by (He *et al.*, 2019a). We are using PyCharm IDE for all the implemented codes. Pruning and fine-tuning was done in a combination of CPUs and GPU in both filter selection and fine-tuning. The characteristics of the used system is as follows

System Configuration:

- Operating System: Windows 10 (Version 10.0.18363, Release 10)
- CPU: Intel64 Family 6 Model 158 Stepping 13, GenuineIntel
- Physical Cores: 16
- Logical Cores: 16
- Total RAM: 64 GB

Python Configuration:

- Python Version: 3.9.12
- Python Compiler: MSC v.1916 64-bit (AMD64)

PyTorch Configuration:

- PyTorch Version: 1.11.0
- CUDA Version: 11.3 (cuDNN 8200)
- GPU: 1 × NVIDIA GeForce RTX 2080 Ti with 11 GB GDDR6 memory

The fine-tuning setup details for results are as follows

Fine-tuning setup for ResNet-50 on Imagenet:

- Training Batch size: 64
- Testing Batch size: 32
- Epochs for fine-tuning: 110
- momentum = 0.9
- Optimizer for fine-tuning: Adam
- weight decay=0.0001
- Number of workers: 4
- Initial learning rate: e-03
- Learning rate schedule: Every 30 epochs
- Pruning rate: {0.2, 0.32, 0.72}

Fine-tuning setup for ResNet-56 on CIFAR-10:

- Training Batch size: 64
- Testing Batch size: 32
- Epochs for fine-tuning: varying between 200 and 300
- momentum = 0.9
- Optimizer for fine-tuning: Adam
- weight decay= 0.0005
- Number of workers: 2
- Initial learning rate: e-03
- Learning rate schedule: 3 times depending on the number of epochs.
- Pruning rate: {0.09, 0.17, 0.25, 0.50, 0.61}

CHAPTER 4

RESOURCE-EFFICIENT AND LAYER INTERDEPENDENCE-AWARE CNN PRUNING LEVERAGING FILTER REPLACEMENT

Sadegh Tofigh[1] , Mohammad Askarizadeh [1] , M. Omair Ahmad [2] , M. N. S. Swamy [2] , Kim Khoa Nguyen [1]

[1] Department of Electrical Engineering, École de Technologie Supérieur, 1100 Notre-Dame Ouest, Montréal, Québec, Canada H3C 1K3

[2] Department Electrical and Computer Engineering, Concordia University, 1455 Blvd. De Maisonneuve Ouest, Montreal, Quebec, Canada H3G 1M8

Article accepted by IEEE Transactions on Neural Networks and Learning Systems, January 2026.

Abstract

Convolutional neural network (CNN) pruning has traditionally relied on heuristically designed importance criteria, often leading to limited generalizability and inconsistent performance. In this paper, we propose a novel framework centered around *Filter Replacement*, introducing pruning as a process of replacing selected filters with zero filters. Through a rigorous analysis, we derive an upper bound on the absolute error in the output of the subsequent layer and use this bound to define an efficient importance function. This importance function exhibits γ-weakly submodular properties, enabling the development of a simple, low-complexity, and data-free oblivious algorithm for selecting filters to prune. In addition, we extend the filter replacement framework to include nonzero filter alternatives, leveraging a best-approximation technique to construct optimal replacements for the pruned filters. Extensive experiments on benchmark networks and datasets validate the effectiveness of our method. The proposed approach achieves state-of-the-art results, with a complexity comparable to basic techniques such as l_2-norm pruning. Notably, our pruning method achieves 76.52% accuracy in ResNet-50 on ImageNet dataset, surpassing the baseline of 75.15%, while reducing network parameters by 25.5%. Our proposed Resource Efficiency (RE) metric assesses that the LIAP method is up to 10^{11} times more efficient than existing techniques, setting a new standard for resource-aware CNN pruning.

Keywords: Convolutional Neural Networks, Filter Pruning, Model Compression Methods, Best Approximation.

4.1 Introduction

Despite their extremely successful applications in computer vision, Convolutional Neural Networks are not well represented on edge devices with limited computing power and memory capacity. Therefore, compression techniques have been proposed to compress and accelerate CNNs by reducing the number of parameters and/or the number of FLOPs in the network, e.g., pruning, parameter quantization, and knowledge distillation (Cheng *et al.*, 2017). Among these approaches, pruning effectively enables model deployment by eliminating redundant CNN filters. A pruning method is fundamentally defined by two primary factors: the parameters (e.g., filters) within the convolutional layers and the dataset utilized for pruning (Cheng *et al.*, 2017). Based on the parameter factor, pruning techniques are broadly categorized into *structured* and *unstructured* approaches. Structured pruning methods include those proposed by (Ganjdanesh *et al.*, 2024), (Li *et al.*, 2023), (Park *et al.*, 2024), (Lee, Ajanthan & Torr, 2018), (Joo *et al.*, 2021), (Lei, Liang, Zheng & Wang, 2022), (Lin *et al.*, 2020), (;and Hans Peter Graf, 2017), (Luo *et al.*, 2018), (Alwani *et al.*, 2021), (Qian *et al.*, 2023), (Lin *et al.*, 2021), (Chen *et al.*, 2024), (Zniyed *et al.*, 2025), (Lee & Song, 2024), and (Tang *et al.*, 2025). In contrast, unstructured pruning methods have been explored in (Ding *et al.*, 2019), (Frankle & Carbin, 2019), and (Han *et al.*, 2015). While unstructured pruning often necessitates specialized hardware for efficient execution, structured pruning is generally preferred due to its compatibility with standard hardware and its effectiveness in enabling efficient model deployment.

Pruning methods can be classified into two categories based on dataset usage. The first category, *data-based* methods, selects filters by analyzing convolutional layer outputs using the dataset during pruning (Ganjdanesh *et al.*, 2024), (Li *et al.*, 2023), (Park *et al.*, 2024), (Lee *et al.*, 2018), (Joo *et al.*, 2021), (El Halabi *et al.*, 2022), (Luo *et al.*, 2018), (Tofigh *et al.*, 2022), (Lin *et al.*, 2020), (Qian *et al.*, 2023), (Lin *et al.*, 2019), (Lee & Song, 2024). These methods evaluate features from either the *current layer* or the *adjacent layer* (He & Xiao, 2024). While adjacent

layer approaches provide a more comprehensive assessment by considering removed parameters from neighboring layers, they also introduce higher computational complexity. The second category, *data-free* methods (;and Hans Peter Graf, 2017), (He *et al.*, 2019b), (Yvinec *et al.*, 2021), (Yvinec *et al.*, 2023), (Chen *et al.*, 2024), (Zniyed *et al.*, 2025), (Tang *et al.*, 2025), selects filters based solely on convolutional layer parameters without using dataset information. These methods are divided into two subgroups: *filter norm* and *filter correlation* (He & Xiao, 2024). While filter norm methods have lower computational complexity, they generally yield lower effective accuracy compared to filter correlation methods. In general, data-based pruning methods outperform data-free methods in accuracy but come at a higher computational cost. Regardless of whether pruning is structured or unstructured, data-free or data-based, most approaches are heuristic and lack theoretical guarantees (Ganjdanesh *et al.*, 2024), (Park *et al.*, 2024), (Joo *et al.*, 2021), (Lei *et al.*, 2022), (Luo *et al.*, 2018), (Lin *et al.*, 2020), (;and Hans Peter Graf, 2017), (He *et al.*, 2019b), (Chen *et al.*, 2024), (Zniyed *et al.*, 2025), (Tang *et al.*, 2025). Additionally, their computational complexity poses challenges, limiting their accessibility to practitioners.

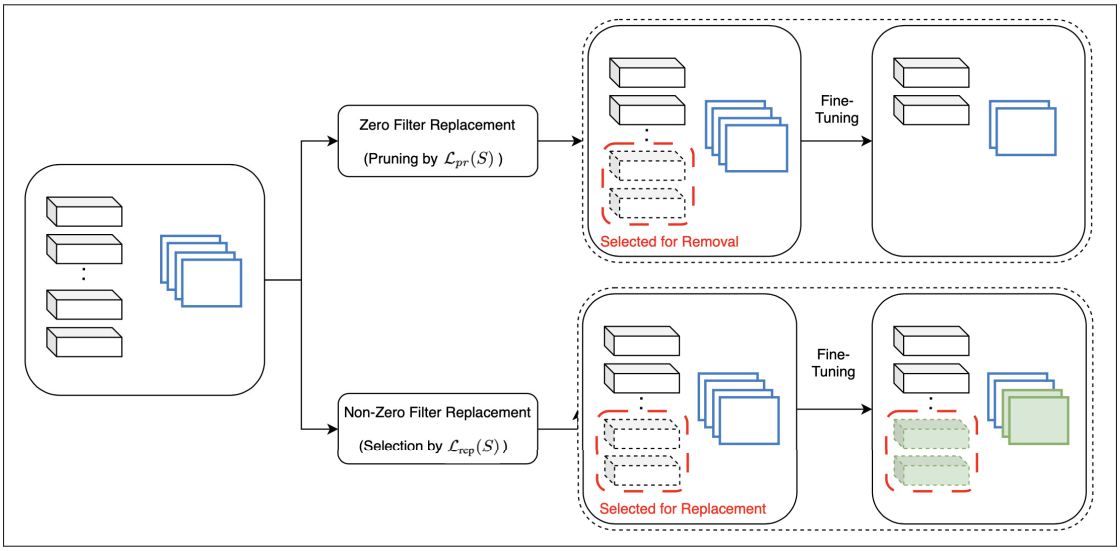

Figure 4.1 Framework of the filter replacement method. In zero filter replacement (pruning), filters are removed from the layer, whereas in non-zero filter replacement, neither the filters nor the output channels are eliminated

In this paper, we address the challenge of CNN model compression by introducing a novel approach centered on filter replacement, which serves as a foundation for our primary contribution: a structured, data-free filter pruning method. While the concept of filter replacement is introduced through a non-zero filter replacement technique based on the best approximation method, its role is to underpin the design of an effective pruning framework. Specifically, we interpret pruning as replacing a filter with a zero filter and develop a pruning strategy guided by a robust theoretical framework, rather than purely empirical heuristics. The proposed pruning approach, termed Layer Interdependence-Aware Pruning (LIAP), extends the filter selection principle of ThiNet (Luo *et al.*, 2018) while mitigating its computational overhead through a novel formulation of an upper bound on the Average Absolute Error (AAE) in the output of the subsequent layer. This bound enables the construction of an efficient importance function that ensures minimal propagation of pruning-induced errors to adjacent layers. By explicitly capturing the interdependence between consecutive convolutional layers, LIAP provides a more informed and balanced pruning process than conventional data-free approaches. In contrast to norm-based methods such as L1 (;and Hans Peter Graf, 2017), which focus solely on intra-layer characteristics, LIAP incorporates cross-layer parameter relationships, resulting in a more comprehensive and effective pruning framework. To further mitigate the computational burden of greedy pruning algorithms, often unsuitable for edge or resource-limited environments, LIAP employs a low-complexity oblivious algorithm, supported by the γ-weak submodularity of its objective function. Extensive experimental results show that LIAP consistently outperforms existing pruning methods, achieving high accuracy while retaining the efficiency and simplicity characteristic of norm-based approaches, thereby proving effective and scalable for on-device model compression.

The key contributions of this paper can be summarized as follows:

- We introduce a novel filter replacement technique for convolutional layers, which incorporates both zero-filter replacements (pruning) and optimal nonzero filter alternatives. Additionally, we provide a method to determine the optimal non-zero replacement for any given filter.

- Importance functions for pruning and non-zero filter replacement are introduced, derived from an upper bound on the average absolute error in the subsequent layer.
- Our pruning method operates exclusively based on filter norms, significantly simplifying its computational complexity to match that of traditional norm-based approaches.
- We develop a low-complexity, rapidly converging oblivious algorithm as an alternative to conventional greedy strategies, leveraging the γ-weakly submodular structure of the pruning framework.
- We empirically validate that our methods effectively restore the accuracy of pruned networks, delivering results comparable to state-of-the-art pruning techniques.
- A novel Resource Efficiency (RE) metric is introduced for evaluating pruning methods, defined as the ratio of network accuracy (ACC) to Multiply-Accumulate Operations (MAC).

4.2 Related Work

4.2.1 Computational Complexity of Pruning Techniques

The complexity of a pruning method primarily depends on its reliance on data and the mechanism used for filter selection. Data-free approaches, such as those proposed in (;and Hans Peter Graf, 2017), (He *et al.*, 2019b), (Lin *et al.*, 2021), and (Chen *et al.*, 2024), generally employ simpler algorithms but tend to achieve lower accuracy compared to data-driven techniques. In contrast, data-based methods, including (Ganjdanesh *et al.*, 2024), (Li *et al.*, 2023), (Park *et al.*, 2024), (Lee *et al.*, 2018), (Joo *et al.*, 2021), (Lin *et al.*, 2020), (Luo *et al.*, 2018), (Lin *et al.*, 2019), (Tofigh *et al.*, 2022), (Alwani *et al.*, 2021), (Qian *et al.*, 2023), and (Huang & Wang, 2018), incorporate training data into the filter selection process, often yielding superior accuracy. Although RED and RED++ (Yvinec *et al.*, 2021), (Yvinec *et al.*, 2023) are data-free and less complex than most data-based methods, their three-stage framework introduces additional computational overhead. Data-free structured pruning can be broadly categorized into filter-norm-based and filter-correlation-based approaches (He & Xiao, 2024). The former, exemplified by (;and Hans Peter Graf, 2017), measures filter importance using norm values, while the latter, such as (He

et al., 2019b), (Yvinec *et al.*, 2021), and (Yvinec *et al.*, 2023), relies on inter-filter correlations. However, existing data-free pruning techniques typically assess filter importance within each layer independently, without considering inter-layer dependencies. This limitation restricts their ability to capture the hierarchical relationships among feature maps. The proposed LIAP method overcomes this shortcoming by modeling parameter interdependence between adjacent layers, enabling a more comprehensive and principled filter selection process.

4.2.2 Theoretical Basis of Network Pruning

Most existing pruning methods are heuristic and lack a solid theoretical foundation. Several studies, including (Sarvani *et al.*, 2022), (Guo *et al.*, 2023), (Wang *et al.*, 2020), (Dai *et al.*, 2018), and (Zheng *et al.*, 2021), have explored theoretical formulations based on *mutual information* (MI) or the *information bottleneck* (IB) principle (Naftali *et al.*, 2000), (Tishby & Zaslavsky, 2015). However, estimating MI or IB in high-dimensional spaces is computationally expensive and often impractical. To mitigate this, (Guo *et al.*, 2023) reformulates the IB objective using a Hilbert–Schmidt independence criterion lasso, but the resulting filter selection remains computationally demanding. Other approaches, such as (Wang *et al.*, 2020), (Dai *et al.*, 2018), and (Sarvani *et al.*, 2022), require specialized fine-tuning to recover accuracy after pruning, while the framework in (Zheng *et al.*, 2021) still depends on heuristic metrics like the l_1-norm. In related work, (El Halabi *et al.*, 2022) proposed a structured pruning method for limited-data settings based on reweighting (Mariet & Sra, 2016) and submodular optimization, and (Yvinec *et al.*, 2023) derived an upper bound on weight differences to enable efficient filter clustering.

In this work, we propose an approach that defines an importance function based on an upper bound of the AAE in the output of the adjacent layer. The γ-weak property of this function guarantees the convergence of the oblivious algorithm employed for filter selection.

4.2.3 Network Pruning Based on Greedy and Oblivious Algorithms

Greedy algorithms are widely used in pruning methods to approximate optimal solutions (Luo *et al.*, 2018), (Guo *et al.*, 2023), (Lee & Song, 2024). Although effective for NP-hard problems, their iterative nature makes them computationally demanding and less suitable for resource-constrained devices. To overcome this limitation, several studies have investigated oblivious algorithms (Tofigh *et al.*, 2022), (Lin *et al.*, 2020), (;and Hans Peter Graf, 2017), though often without formal theoretical support. Despite the weaker mathematical foundation, oblivious approaches have shown comparable performance to greedy ones. For example, F-ThiNet (Tofigh *et al.*, 2022), adopted an oblivious algorithm instead of a greedy strategy and achieved similar accuracy with substantially lower complexity.

In this work, we extend this idea by employing an oblivious algorithm to design a low-complexity pruning framework optimized for resource-limited environments.

4.2.4 Filter Selection Approaches in Structured Pruning Methods

Structured pruning methods derive their importance functions from diverse perspectives, each emphasizing different components of convolutional neural networks (He & Xiao, 2024). Weight-dependent approaches determine filter importance solely based on the parameters within convolutional layers. For instance, (;and Hans Peter Graf, 2017) employs filter norms as an importance measure, whereas FPGM (He *et al.*, 2019b) and RED++ (Yvinec *et al.*, 2023) compute importance through median-based and multi-stage pruning strategies, respectively. Conversely, several methods incorporate dataset information into the importance estimation. ThiNet (Luo *et al.*, 2018) and F-ThiNet (Tofigh *et al.*, 2022) select filters by approximating the absolute error in the output of the adjacent layer, while HRank (Lin *et al.*, 2020) measures filter importance based on the rank of feature maps. Similarly, CHIP (Sui *et al.*, 2021) and NISP (Yu *et al.*, 2018) assess importance using activation statistics obtained during forward propagation. In addition, regularization-based approaches such as NS (Liu *et al.*, 2017), GBN (You *et al.*, 2019), PR (Zhuang *et al.*, 2020), RSNLI (Ye *et al.*, 2018), SSS (Huang & Wang,

2018), and GAL (Lin *et al.*, 2019) enforce structured sparsity by applying regularizers to batch normalization or auxiliary parameters. Other methods, including RLAL (Ganjdanesh *et al.*, 2024) and DTP (Li *et al.*, 2023), perform joint optimization of pruning and network training using reinforcement learning or optimal transport formulations. Optimization-based pruning frameworks also provide alternative means of defining filter importance. For example, (Joo *et al.*, 2021) and (Lei *et al.*, 2022) determine optimal linear combinations of filters via optimization, while (Peng *et al.*, 2019), (Wang *et al.*, 2019), and (Nonnenmacher *et al.*, 2022) approximate the loss function through Taylor expansion to evaluate the contribution of individual filters. In this work, we propose a weight-dependent importance function that considers the norms of filters in both the pruned and adjacent layers. The proposed LIAP importance function, inspired by ThiNet (Luo *et al.*, 2018) and RED++ (Yvinec *et al.*, 2023), minimizes an upper bound of the AAE in the output of the adjacent layer, thereby providing a theoretically grounded and computationally efficient pruning criterion.

4.3 Filter Replacement (FR) in a Convolutional layer

Replacing a filter in a convolutional layer with another does not inherently reduce the total number of network parameters or computational operations. However, under specific conditions, such replacements can substantially reduce computational complexity. In this section, we analyze the process of replacing a filter in a convolutional layer with an alternative, focusing on its effect on the absolute error in the output of the $(l+1)^{\text{st}}$ layer when a filter from layer l is substituted. Our analysis demonstrates that, under certain conditions, filter replacement can effectively reduce the network's computational cost. To this end, we propose two approaches: a zero-filter replacement (pruning) method and an optimal nonzero filter replacement strategy, leveraging the linear combination of filters for pruning, see Figure 4.1.

In the l^{th} convolutional layer of a CNN, the set of filters, $F^l_{[:,:,:,:]}$, consists of H_l filters, each with dimensions $k_l \times k_l \times H_{l-1}$, where H_l denotes both the number of filters in this layer and, equivalently, the number of output channels. The vector space spanned by the filters in the l^{th} convolutional layer, $\text{Span}(F^l_{[:,:,:,:]})$, is a subspace of $\mathbf{T}_{k_l \times k_l \times H_{l-1}}$, the space of tensors of shape

$k_l \times k_l \times H_{l-1}$. If the p^{th} filter in the l^{th} layer, $F^l_{[:,:,:,p]}$, can be expressed as a linear combination of a subset of size K from the remaining filters, i.e.,

$$F^l_{[:,:,:,p]} = \sum_{\substack{r=1 \\ r \neq p}}^{K} a_r F^l_{[:,:,:,r]}, \tag{4.1}$$

then the corresponding p^{th} channel in the output feature map of the l^{th} layer, $X_l[:,:,p]$, can likewise be represented as a linear combination of the other output channels using the same coefficients $a_1, \ldots, a_K$, as follows:

$$X_l[:,:,p] = \sum_{\substack{r=1 \\ r \neq p}}^{K} a_r X_l[:,:,r]. \tag{4.2}$$

Setting all coefficients $a_1, \ldots, a_K$ to zero yields a zero filter, effectively removing the p^{th} filter from the layer. Furthermore, the following theorem establishes a sufficient condition under which a generator set $\mathcal{S} \subset F^l_{[:,:,:,:]}$ can be used to reduce computational complexity in the convolutional layer through structured filter replacement.

Theorem 5. *Let $\mathcal{S}$ be a subset of the filters in the l^{th} convolutional layer of a CNN. If $|\mathcal{S}| < k_l k_l H_{l-1}$, then replacing any filter not in $\mathcal{S}$ with a linear combination of the filters in $\mathcal{S}$ reduces the number of multiplications required to compute the output.*

In a convolutional layer, when the number of filters exceeds the dimensionality of the filter space, i.e., $H_l > k_l k_l H_{l-1}$, it follows that $H_l - k_l k_l H_{l-1}$ filters in the l^{th} layer can be expressed as linear combinations of the others. Consequently, $H_l - k_l k_l H_{l-1}$ output channels are also linearly dependent and can be represented as linear combinations of the remaining channels. In such cases, a subset $\mathcal{S} \subset F^l_{[:,:,:,:]}$ containing $k_l k_l H_{l-1}$ linearly independent filters spans the entire tensor space $\mathbf{T}_{k_l \times k_l \times H_{l-1}}$. Computing each output entry either by direct convolution or by linear combinations of outputs from filters in $\mathcal{S}$ involves the same computational complexity, namely $k_l k_l H_{l-1}$ multiplications. To achieve an actual reduction in computational cost, the size

of the generator set $\mathcal{S}$ must be strictly less than $k_l k_l H_{l-1}$. A detailed example is provided in the supplementary materials.

To analyze the impact of filter replacement on the performance of a CNN, the subsequent theorem defines an upper bound on the AE of the subsequent layer's output when a filter in the l^{th} layer is replaced with an alternative filter.

Theorem 6. *Consider a CNN where the p^{th} filter in the l^{th} convolutional layer, $F^l_{[:,:,:,p]}$, is replaced with an alternative filter $\hat{F}^l_{[:,:,:,p]}$. The AE in the output of the $(l+1)^{st}$ layer caused by this replacement is bounded by:*

$$AE \leq \|F^{l+1}_{[:,:,p,:]}\|_2^2 \cdot \|F^l_{[:,:,:,p]} - \hat{F}^l_{[:,:,:,p]}\|_2^2 \cdot \Lambda^{l-1}, \tag{4.3}$$

where Λ^{l-1} is a constant that depends on the statistical properties of the dataset and the parameters of all preceding layers up to $(l-1)$. It is independent of the parameters of the l^{th} layer.

Proof. The detailed proof is provided in section 4.8.2 □

Theorem 6 identifies the key factors affecting the AE in the output of the $(l+1)^{\text{st}}$ layer when the p^{th} filter in the l^{th} layer is replaced. The AE is influenced by two primary factors:

1. The L_2-norm of the difference between the original filter, $F^l_{[:,:,:,p]}$, and the replacement filter, $\hat{F}^l_{[:,:,:,p]}$.
2. The L_2-norm of the p^{th} channels of the filters in the $(l+1)^{\text{st}}$ layer, denoted as $F^{l+1}_{[:,:,p,:]}$.

These factors collectively determine the extent to which the replacement affects the network's output.

Selecting k filters for replacement from the l^{th} convolutional layer ($l = 1, \ldots, \mathbb{L}$), which contains H_l filters, based on their minimum importance is equivalent to retaining the $H_l - k$ filters with the highest importance scores. Consequently, the task of identifying the $H_l - k$ most significant

filters to preserve in the l^{th} layer can be expressed as the following optimization problem:

$$\max_{\substack{S\subset\{1,\ldots,H_l\}\\ |S|=H_l-k}} \mathcal{L}(S), \tag{4.4}$$

where $\mathcal{L}(S)$ denotes the importance of the filters identified by the index set S.

4.3.1 Filter Pruning, Replacing a Filter with Zero-Filter

We define $\mathcal{L}_{pr}$, the importance function of our data-free LIAP method, on a subset $S \subset \{1, \ldots, H_l\}$ of indices as follows:

$$\mathcal{L}_{pr}(S) := \max\{\| F^l[:,:,:,S] \|_2^2, \| F^{l+1}[:,:,S,:] \|_2^2\}, \tag{4.5}$$

where $F^l[:,:,:,S]$ and $F^{l+1}[:,:,S,:]$ represent the tensor created by filters in the l^{th} layer with indices in S and the tensor of the channels of the filters in $(l+1)^{st}$ layer with indices from S, respectively.

Note: For every set $S \subset V$ of size k, the following inequalities are hold for importance function $\mathcal{L}_{pr}$ defined in (4.5)

$$\mathcal{L}_{pr}(S) \leq \sum_{i\in S} \mathcal{L}_{pr}(i), \qquad \frac{\sum_{i\in S} \mathcal{L}_{pr}(i)}{k} \leq \mathcal{L}_{pr}(S). \tag{4.6}$$

The function $\mathcal{L}_{pr}$ is designed to model the interaction between the target consecutive convolutional layers and is constructed to minimize its influence on network performance. To illustrate this minimal impact, a modified form of Theorem 7 establishes an upper bound on AE in output of the adjacent layer when the p^{th} filter of the l^{th} layer is replaced by a zero-valued filter of the same dimensions.

Theorem 7. *In a CNN, let X_{l+1} denote the output of the $(l+1)^{st}$ convolutional layer and $\hat{X}^p_{l+1}$ be the output when p^{th} filter from the l^{th} layer ($l = 1, \ldots, \mathbb{L}$) is pruned. The absolute error in*

the output of the $(l+1)^{st}$ *convolutional layer is bounded above as follows*

$$\| X_{l+1} - \hat{X}_{l+1}^{p} \|_2^2 \leq \| F^{l+1}[:,:,p,:] \|_2^2 \| F^{l}[:,:,:,p] \|_2^2 \, \Lambda_l, \tag{4.7}$$

where Λ_l depends only on the parameters of the convolutional layers preceding the l^{th} layer and on the input dataset.

The upper bound established in Theorem 7 pertains to a specific input instance, indicating that the absolute error in the output of the $(l+1)^{\text{st}}$ convolutional layer varies across different inputs. To obtain a data-level generalization, we define the AAE of the $(l+1)^{\text{st}}$ layer output, denoted as $AAE(X_{l+1})$, as:

$$\begin{aligned} AAE(X_{l+1}) &= E(\| X_{l+1} - \hat{X}_{l+1}^{p} \|_2^2) \\ &\leq \| F^{l+1}[:,:,p,:] \|_2^2 \| F^{l}[:,:,:,p] \|_2^2 \, E(\Lambda_l). \end{aligned} \tag{4.8}$$

Equation (4.8) establishes the theoretical justification for the formulation of the importance function $\mathcal{L}_{pr}$ introduced in (4.5). This formulation ensures that when the larger of the two norms, $\|F^{l+1}[:,:,p,:]\|_2^2$ and $\|F^{l}[:,:,:,p]\|_2^2$, for the p^{th} filter is small, both quantities remain minimized, thereby tightening the bound in Equation (4.8). As a result, removing such filters introduces the least AAE in the output of the $(l+1)^{\text{st}}$ convolutional layer. For simplicity, we hereafter denote F_{SF}^{l} and F_{SC}^{l+1} as $F^{l}[:,:,:,S]$ and $F^{l+1}[:,:,S,:]$, respectively.

4.3.2 An Optimum Nonzero Filter Replacement

The first step in the non-zero filter replacement process involves selecting filters for replacement using the following importance function:

$$\mathcal{L}_{\text{rep}}(S) = \|F_{SC}^{l+1}\|_2^2. \tag{4.9}$$

This function directly targets one of the dominant factors contributing to the upper bound of the AE in the $(l+1)^{\text{st}}$ layer, as established in Theorem 6. By prioritizing filters with a significant impact on AE, this selection method ensures that replacements are focused on minimizing computational error. The remainder of this section details the methodology for achieving an optimal nonzero replacement for the selected filters, balancing computational efficiency with network performance.

Designing the Optimal Replacement Filter: To minimize the absolute error as defined in Equation (4.3), the next step in the non-zero filter replacement process employs the best approximation method. This method identifies the optimal alternative filter by minimizing its distance to the filters in the set S, ensuring that the replacement is a linear combination of these filters. According to Theorem 4.9 in (Deutsch & Deutsch, 2001), which characterizes the best approximation of a vector from a subspace, the best approximation of the filter $F^l_{\{p\}T}$ within the vector subspace spanned by $\mathcal{S} = \left\{F^l_{\{p_j\}T}\right\}_{j=1,\ldots,K}$ can be expressed as:

$$\hat{F}^l_{\{p\}T} = \mathcal{T}_{k_l,k_l,H_l}(\mathcal{M}\Gamma_p), \tag{4.10}$$

where: $\mathcal{T}_{k_l,k_l,H_l}$ transforms a vector of size $k_l k_l H_l$ into a tensor of size $k_l \times k_l \times H_l$, $\mathcal{M}$ is a $(k_l k_l H_l) \times K$ matrix, with its columns as the vectorized forms of the filters in S, Γ_p is a $K \times 1$ column vector computed as:

$$\Gamma_p = \mathcal{G}^{-1}\beta, \tag{4.11}$$

where $\mathcal{G}$ is the symmetric $K \times K$ Gram matrix of $\mathcal{S}$, and β is a $K \times 1$ real-valued column vector defined as:

$$\beta = \left[\left\langle F^l_{\{p\}F}, F^l_{\{p_1\}F}\right\rangle, \ldots, \left\langle F^l_{\{p\}F}, F^l_{\{p_K\}F}\right\rangle\right]^t. \tag{4.12}$$

In our proposed filter replacement method, the filter $F^l_{\{p\}T}$ virtually substitute with $\hat{F}^l_{\{p\}T}$. After calculating the vector Γ_p, the filter $F^l_{\{p\}T}$ can be excluded from the network, while the output channel p is reconstructed as a linear combination of the output channels $p_1, p_2, \ldots, p_K$, with

the corresponding coefficients from Γ_p:

$$X_l[:,:,p] = \sum_{j=1}^{K} \Gamma_p[j] X_l[:,:,p_j]. \tag{4.13}$$

This method assumes the linear independence of the filters within the set $\mathcal{S}$. In the following section, we analyze the weak submodularity of the pruning problem in relation to the proposed importance functions $\mathcal{L}_{\text{rep}}$ and $\mathcal{L}_{\text{pr}}$.

4.4 Proposed Filter Selections as Weakly Submodular Maximization Problems

In this section, we prove that the importance function $\mathcal{L}_{pr}$ exhibits γ-weak submodularity, while $\mathcal{L}_{rep}$ is strictly submodular. Let $V = \{1, 2, \ldots, d\}$ denote the ground set, and define a set function $\mathcal{F} : 2^V \to \mathbb{R}_+$. The marginal gain from adding a subset $J \subseteq V$ to another subset $R \subseteq V$ is expressed as $F(J \mid R) = \mathcal{F}(R \cup J) - \mathcal{F}(R)$, which quantifies the incremental change in $\mathcal{F}$ due to the inclusion of J. The size of R is denoted by $|R|$.

Definition 4.1 ((El Halabi *et al.*, 2022))**.** *A set function $\mathcal{F} : 2^V \to \mathbb{R}_+$ is said to be $\gamma_{U,k}$-weakly submodular for a subset $U \subseteq V$ and integer $k \in \mathbb{N}$ if there exists $\gamma_{U,k} > 0$ such that*

$$\gamma_{U,k}\, \mathcal{F}(R \mid L) \leq \sum_{i \in R} \mathcal{F}(i \mid L), \tag{4.14}$$

for all disjoint $L, R \subseteq V$ satisfying $L \subseteq U$ and $|R| \leq k$. The constant $\gamma_{U,k}$ is referred to as the submodularity ratio of $\mathcal{F}$.

A widely adopted strategy for addressing weakly submodular optimization problems is the greedy algorithm (Das & Kempe, 2011). Given the ground set V, a subset $S \subseteq V$ is constructed iteratively by selecting k elements. The process begins with an empty set and successively adds the index that maximizes $\mathcal{F}(S \cup \{i\})$ at each step, continuing until $|S| = k$. This procedure ensures that, at every iteration, the element offering the greatest marginal gain relative to the existing selection is chosen. Although the greedy algorithm provides strong theoretical guarantees and has been extensively applied in pruning frameworks (El Halabi *et al.*, 2022),

(Luo *et al.*, 2018), its iterative design results in substantial computational overhead, limiting its feasibility for deployment on constrained hardware.

To mitigate the computational burden of iterative selection, we introduce the oblivious algorithm, a streamlined alternative to the greedy approach. Rather than performing successive updates, the oblivious algorithm directly identifies a subset S^{obv} of k indices with the highest scores among $\{\mathcal{F}(1), \ldots, \mathcal{F}(d)\}$, as determined in the evaluation phase of the greedy procedure, i.e.,

$$\sum_{i \in S^{obv}} \mathcal{F}(i) \geq \sum_{i \in S} \mathcal{F}(i), \tag{4.15}$$

for all $S \subset V$ of size k.

Theorem 8 confirms the submodularity of the LIAP importance function, allowing the use of an oblivious algorithm with provable approximation guarantees.

Theorem 8. *The importance functions $\mathcal{L}_{rep}$ and $\mathcal{L}_{pr}$ defined in Equations* (4.9) *and* (4.5) *are submodular function and $\gamma_{U,k}$-weakly submodular function, respectively.*

Proof. The detailed proof is provided in section 4.8.3. □

Note: In this work, we assume that for each filter $F^l_{\{i\}F}$ in the l^{th} convolutional layer, at least one of the norms $\|F^l_{\{i\}F}\|_2^2$ or $\|F^{l+1}_{\{i\}C}\|_2^2$ is nonzero. Filters with both norms equal to zero are excluded from consideration, as they have no contribution to the network's output.

Note: Let S^* denote the optimal subset of k filters in (4.4). For the importance function $\mathcal{L}_{pr}$ in (4.5), the following holds:

$$\gamma_{\emptyset,k} = \min_{R \subset V, |R| \leq k} \frac{\sum_{i \in R} \mathcal{L}_{pr}(i)}{\mathcal{L}_{pr}(R)} \geq 1, \tag{4.16}$$

$$\gamma_{\emptyset,k} \leq \frac{\sum_{i \in S^*} \mathcal{L}_{pr}(i)}{\mathcal{L}_{pr}(S^*)}. \tag{4.17}$$

Computing $\gamma_{\emptyset,k}$ requires solving an NP-hard combinatorial problem over all filter subsets of size up to k, amounting to $\sum_{i=0}^{k} \binom{H_l}{i}$ possibilities. For example, the first layer of VGG-16 with 64 filters yields over 8 million subsets when $k = 5$. As shown in Theorem 8, the proposed importance function $\mathcal{L}_{pr}$ is γ-weakly submodular. Accordingly, the subsequent section employs the oblivious algorithm to solve this maximization efficiently, whose convergence for weakly submodular functions is guaranteed in (Das & Kempe, 2011).

4.5 Oblivious Algorithm For The Proposed Pruning Method

In this section, we employ the oblivious algorithm to tackle the optimization problem (4.4). Subsequently, we establish in Theorem 9 that the $\gamma_{U,k}$-weak submodularity of the importance function $\mathcal{L}_{pr}$, as defined in (4.5), guarantees that the solution obtained by this algorithm attains a near-optimal solution, with its deviation from the optimum bounded by $\frac{\gamma_{\emptyset,k}}{k}$, where $\emptyset$ denotes the empty set. Algorithm 4.1 outlines the oblivious procedure employed in the proposed LIAP method to select the set S^{obv} of filters to be pruned from the l^{th} convolutional layer.

Algorithm 4.1 Oblivious algorithm for selecting filters to prune from the l^{th} convolutional layer

Input: The set of filters in the l^{th} and $(l+1)^{st}$ convolutional layers. Pruning rate r;
Output: S^{obv}, the set of selected filters in the l^{th} layer for removal;

$I \leftarrow \{1, 2, \ldots, H_l\}$;
$S^{obv} \leftarrow \emptyset$;
Compute $\hat{\mathcal{L}}(p) = \max(\| F^{l}_{\{p\}F} \|_2^2, \| F^{l+1}_{\{p\}C} \|_2^2)$ for each $p \in \{1, \ldots, H_l\}$;
while $|S^{obv}| \leq r \cdot H_l$ **do**
    Compute the value of index p^* using Equation (4.18);
    Add p^* to index set S^{obv};
    Remove p^* from I;
end while

4.5.1 Algorithm Procedure

The importance of each filter in the l^{th} convolutional layer is computed by first evaluating the ℓ_2-norms of the filter $F^{l}_{\{p\}F}$ and its corresponding output channels $F^{l+1}_{\{p\}C}$ for all $p = 1, \ldots, H_l$.

The importance value $\mathcal{L}_{pr}(p)$ is then defined as the larger of the two norms, $\|F^l_{\{p\}F}\|_2^2$ and $\|F^{l+1}_{\{p\}C}\|_2^2$. The filter associated with the smallest $\mathcal{L}_{pr}(p)$, denoted by index p^*, is identified as the least important and selected for removal from the l^{th} layer.

In general, the index of the i^{th} filter to be removed, p_i^*, is identified by

$$p_i^* = \arg\min_{p \in I} \mathcal{L}_{pr}(p), \tag{4.18}$$

where $i = 1, \ldots, k$, and $I = \{1, \ldots, H_l\} \setminus \{p_1^*, \ldots, p_{i-1}^*\}$. The following theorem demonstrates that Algorithm 4.1 achieves an approximation bound expressed in terms of $\gamma_{\emptyset,k}$.

Theorem 9. *The set S^{obv} of size k selected by the oblivious algorithm has the following approximation guarantees:*

$$\mathcal{L}_{pr}(S^{obv}) \geq \frac{\gamma_{\emptyset,k}}{k} \mathcal{L}_{pr}(S^*), \tag{4.19}$$

and

$$\mathcal{L}_{pr}(S^{obv}) + \mathcal{R}_{obv} \geq \gamma_{\emptyset,k} \mathcal{L}_{pr}(S^*), \tag{4.20}$$

where S^* is the optimal solution of the problem (4.4)*, and $\mathcal{R}_{obv}$ is defined as $\| F^l_{S_1^{obv}F} \|_2^2 - \| F^{l+1}_{S_1^{obv}C} \|_2^2$ or $\| F^{l+1}_{S_2^{obv}C} \|_2^2 - \| F^l_{S_2^{obv}F} \|_2^2$, depending on whether $\mathcal{L}_{pr}(S^{obv})$ is given by $\| F^l_{S^{obv}F} \|_2^2$ or $\| F^{l+1}_{S^{obv}C} \|_2^2$, respectively, and the sets S_1^{obv} and S_2^{obv} are defined as follows:*

$$S_1^{obv} = \{i \in S \mid\mid \| F^l_{\{i\}F} \|_2^2 \geq \| F^{l+1}_{\{i\}C} \|_2^2\}, \tag{4.21}$$

and

$$S_2^{obv} = \{i \in S \mid\mid \| F^l_{\{i\}F} \|_2^2 \leq \| F^{l+1}_{\{i\}C} \|_2^2\}. \tag{4.22}$$

Proof. The detailed proof is provided in section 4.8.4. □

Equation (4.19) shows that the index set selected by the oblivious algorithm approximates the optimal set S^* within a factor of $\frac{\gamma_{\emptyset,k}}{k}$. Similarly, Equation (4.20) expresses this approximation in terms of $\gamma_{\emptyset,k}$ and $\mathcal{R}_{obv}$; the closer $\gamma_{\emptyset,k}$ is to 1 and $\mathcal{R}_{obv}$ is to 0, the nearer the solution is to

optimal. For instance, if in layer l of a CNN each filter norm equals that of its corresponding channels in layer $l+1$, then $\gamma_{\emptyset,k} = 1$ and $\mathcal{R}_{obv} = 0$, meaning the oblivious algorithm attains the exact optimal solution. The following theorem provides more examples.

As follows, we explore the submodularity of $\| \ . \ \|_2^2$ as a set function on the set of filters in a convolutional layer. For the set of filters in the l^{th} layer of a CNN, the function $\| \ . \ \|_2^2$ can be regarded as a set function. Given a subset S of $\{1, 2, \ldots, H_l\}$, the index set of the filters in the l^{th} layer, it returns $\| \ F^l[:, :, :, S] \ \|_2^2$. The following lemma can be easily proven for this set function.

Lemma 2. *For any subset $S \subset \{1, 2, \ldots, H_l\}$ and any index $\alpha \notin S$ we have:*

$$\| \ F^l[:, :, :, S \cup \{\alpha\}] \ \|_2^2 - \| \ F^l[:, :, :, S] \ \|_2^2 = \| \ F^l[:, :, :, \alpha] \ \|_2^2, \tag{4.23}$$

and

$$\begin{aligned} &\| \ F^{l+1}[:, :, S \cup \{\alpha\}, :] \ \|_2^2 - \| \ F^{l+1}[:, :, S, :] \ \|_2^2 = \\ &\| \ F^{l+1}[:, :, \alpha, :] \ \|_2^2 \ . \end{aligned} \tag{4.24}$$

The following theorem shows that $\| \ . \ \|_2^2$ is a submodular set function.

Theorem 10. *The submodularity ratio of the set function $\| \ . \ \|_2^2$ is 1. The set of size K selected by the oblivious algorithm equipped with the $\| \ . \ \|_2^2$, as the importance function, is the optimal solution.*

Proof: Let's assume $U \subset V$. For any $L \subset U$ and $S \subset V$ where $|S| \leq k$

$$\gamma_{U,k} = \min_{L \subset U \ S \subset V, |S| \leq k} \left(\frac{\sum_{i \in S} \| \ i|L \ \|_2^2}{\| \ S|L \ \|_2^2} \right) \tag{4.25}$$

$$= \min_{L \subset U \ S \subset V, |S| \leq k} \left(\frac{\sum_{i \in S} \| \ F^l_{(L \cup \{i\})F} \ \|_2^2 - \| \ F^l_{LF} \ \|_2^2}{\| \ F^l_{(L \cup \{S\})F} \ \|^2 - \| \ F^l_{LF} \ \|^2} \right) \tag{4.26}$$

$$= \min_{L \subset U \ S \subset V, |S| \leq k} \left(\frac{\sum_{i \in S} \| \ F^l_{\{i\}F} \ \|_2^2}{\| \ F^l_{SF} \ \|_2^2} \right) = 1. \tag{4.27}$$

This theorem shows that when the filter selection criteria is only the l_2-norm of the filters, the $\gamma_{U,k} = 1$. It means that the $\| \ . \ \|_2^2$ is a submodular set function . On the other hand, since

$$\sum_{i \in S} \| \ F^l_{\{i\}F} \ \|_2^2 = \| \ F^l_{SF} \ \|_2^2, \tag{4.28}$$

a set S of filters has the maximum l_2-norm when the filters in S individually have the greatest norms among the filters in the l^{th} layer. Therefore, the oblivious algorithm equipped with $\| \ldots \|_2^2$ provides the optimal solution to the maximization problem (4.4).

Remark 3. *In a convolutional layer, if* $\| \ F^l_{\{i\}F} \ \|_2^2 \geq \| \ F^{l+1}_{\{i\}C} \ \|_2^2$ *for all* $i = 1, \ldots, H_l$*, then the importance function* $\mathcal{L}_{pr}$ *is submodular, and* $S^{obv} = S^*$.

Remark 4. *In a convolutional layer, if* $\| \ F^l_{\{i\}F} \ \|_2^2 \leq \| \ F^{l+1}_{\{i\}C} \ \|_2^2$ *for all* $i = 1, \ldots, H_l$*, then the importance function* $\mathcal{L}_{pr}$ *is submodular, and* $S^{obv} = S^*$.

4.5.2 Influence of Filter and Channel Norms as Primary Factors in the Importance Function $\mathcal{L}_{pr}$

In this section, we examine the influence of the two primary components, $\|F^l_{\{p\}T}\|_2^2$ and $\|F^{l+1}_{\{p\}C}\|_2^2$, on the proposed importance function $\mathcal{L}_{pr}$ from two perspectives: their roles in filter selection and the ablation analysis of their effects.

Roles of the factors in filter selection: To illustrate the contribution of both terms $\|F^l_{\{p\}T}\|_2^2$ and $\|F^{l+1}_{\{p\}C}\|_2^2$, we analyze the distribution of filters whose importance is evaluated based

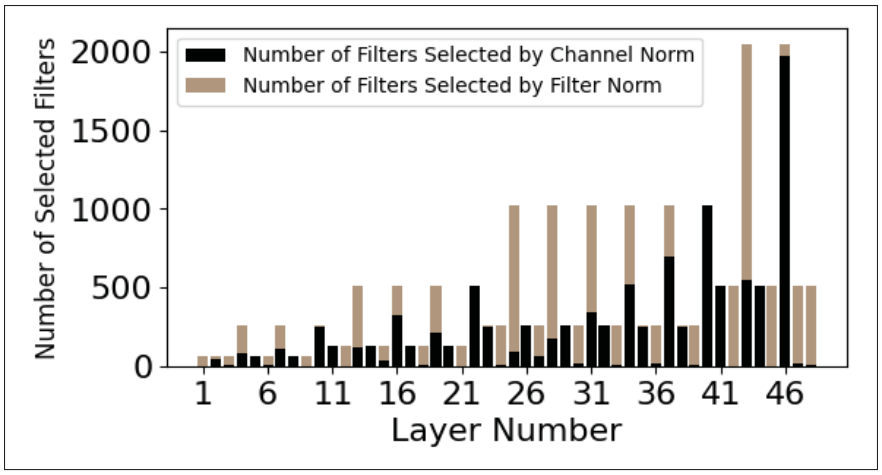

Figure 4.2 Layer-wise analysis of the contributions of $F^{l}_{\{i\}T}$ and $F^{l+1}_{\{i\}C}$ to filter selection by $\mathcal{L}_{pr}$ using the oblivious algorithm. The stacked bars depict the proportion of filters selected based on the norms of $F^{l}_{\{i\}T}$ and $F^{l+1}_{\{i\}C}$

on by $\|F^{l}_{\{p\}T}\|_2^2$ (i.e., $\| F^{l}_{\{p\}T} \|_2^2 \geq \| F^{l+1}_{\{p\}C} \|_2^2$) versus those determined by $\|F^{l+1}_{\{p\}C}\|_2^2$ (i.e., $\| F^{l+1}_{\{p\}C} \|_2^2 \geq \| F^{l}_{\{p\}T} \|_2^2$). Figure 4.2 depicts the layer-wise comparison for ResNet-50 models pre-trained on the ImageNet dataset, showing how frequently each norm dominates in determining filter importance. For example, as an illustration, approximately 75% of filters in the second layer of ResNet-50 derive their importance based on the norms of their associated channels in the third layer, $F^{3}_{\{p\}C}$. Figure 4.2 indicates that the two norms in $\mathcal{L}_{pr}$ jointly influence the evaluation of filter importance.

Ablation study: Furthermore, to assess the contribution of each component in the proposed importance function, we performed an ablation study by isolating the effects of the filter ℓ_2-norm and the ℓ_2-norm of the corresponding output channels in the next layer. Removing the first component reduces our method to the L1 approach (;and Hans Peter Graf, 2017), which, as shown in the experiments, is outperformed by LIAP. Table 4.1 further reports results when filters are selected solely based on the ℓ_2-norm of their output channels. Using a 50% pruning ratio, VGG-11 and AlexNet were fine-tuned for 3 and 15 epochs, respectively. The results demonstrate

that while each component independently contributes to pruning performance, their combination in LIAP consistently achieves higher accuracy across networks and datasets.

Table 4.1 Ablation Study of the LIAP Importance Function

Network	**Dataset**	**LIAP ACC (%)**	**Ablation ACC (%)**
VGG-11	MNIST	99.54%	99.42%
AlexNet	CIFAR-10	84.81%	84.59%

Table 4.2 Formulas for estimating the computational complexity of selecting k filters in the l^{th} convolutional layer among various pruning methods

Pruning Method	**Complexity**
LIAP-Oblivious	$\approx H_l(n_l^1 + m_l^1)$
LIAP-Greedy	$\approx \sum_{i=1}^{k}(H_l - i + 1)[n_l^i + m_l^i]$
RED++ (Yvinec *et al.*, 2023)	$\approx \binom{H_l}{2} n_l^1$
HRank (Lin *et al.*, 2020)	$\approx \mathcal{B}\left(\left(\sum_{i=1}^{l-1} C_{i,H_i}\right) + H_l R_l\right)$
F-ThiNet (Tofigh *et al.*, 2022)	$\approx \mathcal{B} H_l N_{l,1,\mathbb{I}}$
ThiNet (Luo *et al.*, 2018)	$\approx \mathcal{B}\sum_{i=1}^{k}(H_l - i + 1)N_{l,i,\mathbb{I}}$
APIB (Guo *et al.*, 2023)	$\approx \left[\left(\mathcal{B}\sum_{i=1}^{l-1} C_{i,H_i}\right) + 2\mathcal{B}^2\mathcal{D}_l + 4\mathcal{B}^3\right]$

4.5.3 Complexity Analysis

We evaluate the complexity of filter selection across pruning methods, using multiplication count as the primary measure of computational cost. Table 4.2 summarizes the complexity analysis, with the following notations: $\hat{r}_l$ denotes the number of filters retained in the l^{th} layer, $\mathcal{B}$ the batch size, and k_l the kernel size of filters in that layer. W_l and L_l represent the width and length of the layer's output, respectively. Among the baselines in Table 4.2, we compare the complexity of the oblivious and greedy algorithms using the LIAP importance function $\mathcal{L}_{pr}$—the former being our proposed method and the latter serving as a baseline. Additional baselines include RED++ (Yvinec *et al.*, 2023), HRank (Lin *et al.*, 2020), F-ThiNet (Tofigh *et al.*, 2022), ThiNet (Luo *et al.*, 2018), and APIB (Guo *et al.*, 2023), which represent recent state-of-the-art efficient pruning methods.

In the formulas given in Table 4.2, $\mathcal{D}_l = 2W_l^2 L_l^2 + 2W_{l+1}^2 L_{l+1}^2 H_{l+1}^2$, $C_{\mu,\nu} = \nu k_\mu^2 H_{\mu-1} L_\mu W_\mu$ ($\mu = 1, \ldots, \mathbb{L}$; $\nu = 1, \ldots, H_\mu$), and $N_{l,\hat{r}_l,\mathbb{I}} = \left(\sum_{i=1}^{l-1} C_{i,H_i}\right) + C_{l,\hat{r}_l} + \hat{r}_l \mathbb{I} k_{l+1}^2$, where $\mathbb{I}$ denotes the number of entries in the output of the $(l+1)^{\text{st}}$ layer used during filter selection. Moreover, $R_l = \sum_{j=1}^{\min(W_l, L_l)-1} (W_l - j)(L_l - j)$, $n_i^j = k_i^2 H_{i-1} j$, and $m_i^j = k_{i+1}^2 H_{i+1} j$.

As shown in Table 4.2, both APIB and ThiNet require significantly more multiplications than the proposed LIAP method. Similarly, HRank, F-ThiNet, and the greedy algorithm using the LIAP importance function $\mathcal{L}_{pr}$ are also more computationally demanding. Among all methods, RED++ is the closest to LIAP in efficiency yet remains more complex. Notably, the conventional ℓ_2-norm-based filter selection approach (;and Hans Peter Graf, 2017) requires only $H_l n_l^1$ multiplications, exhibiting comparable complexity to LIAP.

4.6 Experiments

In this section, we evaluate the performance of the proposed filter replacement and LIAP pruning methods under data-free conditions. To reduce the computational overhead associated with iterative pruning, we adopt a one-shot compression strategy, in which the pre-trained model is pruned in a single pass. After applying the LIAP pruning method, the network is fine-tuned to recover its accuracy, whereas no fine-tuning is performed following non-zero filter replacement.

4.6.1 LIAP Experimental Results

The pruning procedure for the proposed method consists of three main steps:

1. *Filter Selection:* A uniform pruning ratio r is applied to all convolutional layers except the final one. Filters are ranked and selected for removal using Algorithm 4.1, proceeding sequentially up to the penultimate layer.
2. *Pruning:* The least important filters identified in the previous step are simultaneously removed from all applicable layers.
3. *Fine-Tuning:* The pruned network is fine-tuned to recover its original accuracy.

Table 4.3 Comparison of pruning performance and compression for ResNet-50 on ILSVRC-2012 across existing methods

Model	**Parameters**	**FLOPs**	**Top1 (%)**	**Top-5 (%)**
Baseline (He *et al.*, 2016a)	25.5M	4.09B	76.15	92.87
DECORE (Alwani *et al.*, 2021)	22.69M	3.54B	76.31	93.02
NAS (Lee & Song, 2024)	21.56M	2.48B	76.23	92.87
GAL-0.5 (Lin *et al.*, 2019)	21.20M	2.33B	71.95	90.94
PFP (Liebenwein *et al.*, 2020)	20.88M	3.65B	75.91	92.81
GAL-0.5 joint (Lin *et al.*, 2019)	19.31M	**1.84B**	71.80	90.82
SSS-32 (Huang & Wang, 2018)	18.60M	2.82B	74.18	91.91
LIAP	**18.22M**	2.86B	**76.52**	**93.18**
PFP (Liebenwein *et al.*, 2020)	17.82M	2.29B	75.21	92.43
HRank (Lin *et al.*, 2020)	16.15M	2.30B	74.98	92.33
He et al. (He *et al.*, 2017)	-	2.73B	72.30	90.80
FiltDivNet (Lei *et al.*, 2022)	15.62M	1.66B	75.23	92.5
SSS-26 (Huang & Wang, 2018)	15.60M	2.33B	71.82	90.70
GDP-0.6 (Lin *et al.*, 2018)	-	1.88B	71.19	90.71
GDP-0.5 (Lin *et al.*, 2018)	-	**1.57B**	69.58	90.14
GAL-1 (Lin *et al.*, 2019)	14.67M	1.58B	69.88	89.75
FilterSketch (Lin *et al.*, 2021)	14.53M	2.23B	74.68	92.17
LIAP	**14.61M**	2.26B	**75.37**	**92.56**
GAL-1 joint (Lin *et al.*, 2019)	10.21M	1.11B	69.31	89.12
ThiNet-50 (Luo *et al.*, 2018)	8.66M	1.10B	68.42	88.30
HRank (Lin *et al.*, 2020)	8.27M	0.98B	69.10	**89.58**
FilterSketch (Lin *et al.*, 2021)	7.18M	0.93B	69.43	89.23
LIAP	**6.67M**	**0.96B**	**69.58**	89.31

Model efficiency is evaluated using two standard metrics: the number of parameters and the required Floating Point Operations (FLOPs). Experiments are conducted on two benchmark datasets, CIFAR-10 (Krizhevsky *et al.*, 2009) and ILSVRC-2012 (Deng *et al.*, 2009). To examine the effectiveness of the proposed pruning approach, three widely adopted architectures are employed—ResNet-50, ResNet-56, and ResNet-110—with ResNet-50 evaluated on ILSVRC-2012 and the remaining two on CIFAR-10. The performance of the pruned models is assessed in terms of Top-1 and Top-5 classification accuracy. The proposed LIAP method is compared against a comprehensive set of state-of-the-art baselines, including HRank (Lin *et al.*, 2020), RED++ (Yvinec *et al.*, 2023), ThiNet (Luo *et al.*, 2018), GAL (Lin *et al.*, 2019), SSS (Huang & Wang,

Table 4.4 Comparison of pruning performance and compression for ResNet-56 on CIFAR-10 across existing methods

Model	**Parameters**	**FLOPs**	**Top1 (%)**
Baseline (He *et al.*, 2016a)	0.85M	125.49M(0.0%)	93.26
GAL-0.6 (Lin *et al.*, 2019)	0.75M (11.8%)	78.30M (37.6%)	92.98
L1 (;and Hans Peter Graf, 2017)	0.73M (14.1%)	90.90M (27.6%)	93.06
LIAP	**0.71M (16.8%)**	104.15M (17%)	**94.09**
HRank (Lin *et al.*, 2020)	0.71M(16.8%)	88.72M(29.3%)	93.52
FilterSketch (Lin *et al.*, 2021)	0.68M(20.6%)	88.05M(30.43%)	93.65
DECORE (Lin *et al.*, 2019)	0.64M (24.2%)	92.48(26.3%)	93.34
LIAP	**0.59M (30.43%)**	86.85(30.78%)	**93.84**
SOKS (Liu *et al.*, 2022)	0.51M (40%)	81.40M (36%)	93.22
NAS (Lee & Song, 2024)	0.50M (41.1%)	50.13(60.05%)	93.51
FilterSketch (Lin *et al.*, 2021)	0.50M (41.2%)	73.36M (41.5%)	93.19
NISP (Yu *et al.*, 2018)	0.49M (42.4%)	81.00M (35.5%)	93.01
HRank (Lin *et al.*, 2020)	0.49M (42.4%)	62.72M (50.0%)	93.17
DTP (Ganjdanesh *et al.*, 2024)	-	(50.00%)	93.46
HTP-URC (Qian *et al.*, 2023)	**0.47M (44.72%)**	58.65M (46.58%)	**93.55**
FiltDivNet (Lei *et al.*, 2022)	0.45M (47.36%)	54.85M (56.29%)	93.36
REPrune (Park *et al.*, 2024)	-	(60.38%)	93.40
DECORE (Lin *et al.*, 2019)	0.43M (49%)	62.93M (49.9%)	93.26
LIAP	0.42M (49.97%)	62.12M (50.5%)	93.54
GAL-0.8 (Lin *et al.*, 2019)	0.29M (65.9%)	49.99M (60.2%)	90.36
HRank (Lin *et al.*, 2020)	0.27M (68.1%)	32.52M (71.69%)	90.72
QSFM (Wang *et al.*, 2022)	0.25M (71%)	50.62M (60%)	91.88
CHIP (Sui *et al.*, 2021)	0.24M (71.8%)	34.79M (72.3%)	92.05
FilterSketch (Lin *et al.*, 2021)	0.24M (71.8%)	32.47M (74.4%)	91.2
LRF (Joo *et al.*, 2021)	**(74.1 %)**	(73.9 %)	**93.00**
(Tang *et al.*, 2025)	0.22M (74.1%)	34.09M (73.3%)	92.49
LIAP	0.22M (74.12%)	32.18M (74.36%)	92.56
RGP (Chen *et al.*, 2024)	(75.3%)	(74.5%)	91.5
APIB (Guo *et al.*, 2023)	0.14M (83%)	23.85M (81%)	**91.53**
LIAP	**0.136M (84%)**	20.08M (84%)	91.39

2018), L1 (;and Hans Peter Graf, 2017), F-ThiNet (Tofigh *et al.*, 2022), He et al. (He *et al.*, 2017), GDP (Lin *et al.*, 2018), NISP (Yu *et al.*, 2018), DECORE (Alwani *et al.*, 2021), FilterSketch (Lin *et al.*, 2021), LRF (Joo *et al.*, 2021), FiltDivNet (Lei *et al.*, 2022), DTP (Ganjdanesh *et al.*, 2024), and APIB (Guo *et al.*, 2023). For all comparative methods, reported values are taken directly from their respective publications, with unavailable results denoted by "–". The best pruning performance is highlighted in bold.

Table 4.5 Comparison of pruning performance and compression for ResNet-110 on CIFAR-10 across existing methods

Model	**Parameters**	**FLOPs**	**Top1 (%)**
Baseline (He *et al.*, 2016a)	1.7M(0.0%)	252.89M(0.0%)	93.50
L1 (;and Hans Peter Graf, 2017)	1.16M (32.6%)	155M (38.7 %)	93.3
DECORE (Alwani *et al.*, 2021)	1.11M (36%)	163.30M (35%)	93.88
NORTON (Zniyed *et al.*, 2025)	1.08M (38%)	163.00M (35%)	94.85
HRank (Lin *et al.*, 2020)	1.04M(39.4%)	148.70M(41.2%)	94.23
LIAP	0.97M (43.3%)	142.804M (43.3%)	94.2
GAL-0.5 (Lin *et al.*, 2019)	0.95M(44.8%)	130.20M(48.5%)	92.55
HRank (Lin *et al.*, 2020)	0.70M(59.2%)	105.70M(58.2%)	93.36
LIAP	**0.69M**(59.7%)	101.09M(60.0%)	**93.91**
NORTON (Zniyed *et al.*, 2025)	0.59M (65%)	92.99M (35%)	**94.11**
HRank (Lin *et al.*, 2020)	0.53M(68.7%)	79.30M(68.6%)	92.65
LIAP	0.53M (69.2%)	77.284M (69.4%)	93.28

Configuration: All experiments are implemented in Python 3.9.12 using the PyTorch 1.11.0 framework. cuDNN 8200 is employed to accelerate fine-tuning operations, and PyCharm serves as the integrated development environment for all implementations.

For pruning ResNet-50 on ImageNet, we utilize the pre-trained model provided by PyTorch. The network is fine-tuned for 100 epochs with an initial learning rate of 0.001, which is reduced every 30 epochs. For ResNet-56 and ResNet-110 on CIFAR-10, we adopt the pre-trained models from (He *et al.*, 2019b). These networks are fine-tuned for 200–300 epochs with a learning rate that varies between 0.001, 0.0001, and 0.00001, and a batch size of 64 during fine-tuning. The compression performance of the proposed LIAP method on ResNet-50, ResNet-56, and ResNet-110 is presented in Tables 4.3, 4.4, and 4.5. The tables report the original pre-trained model's parameter count, FLOPs, and accuracy as baseline metrics before pruning.

Note that the baseline methods reported in these tables were not trained under a unified training policy. We followed the publicly available implementations of reference methods such as FPGM (He *et al.*, 2019b) and HRank (Lin *et al.*, 2020), which also employ a one-shot pruning strategy. These implementations do not fix random seeds, and our experimental setup follows the same convention to ensure consistency with existing literature.

ResNet-50 on ImageNet: As shown in Table 4.3, the proposed LIAP method attains a Top-1 accuracy of 76.52%, slightly exceeding the baseline (76.52% vs. 76.15%) while achieving comparable FLOPs and parameter reduction to SSS-32. Compared with FilterSketch, LIAP achieves a higher accuracy (75.37% vs. 74.68%) with a notably larger reduction in parameters and nearly identical FLOPs. Even under high pruning ratios, LIAP outperforms GAL-1 joint, ThiNet-50, HRank, and FilterSketch in Top-1 accuracy (69.58% vs. 69.31%, 68.42%, 69.1%, and 69.43%, respectively) while maintaining greater compression. Since RED++ (Yvinec *et al.*, 2023) does not report numerical results for ResNet-50 on ImageNet, we conduct the comparison based on their metric, the ratio of preserved accuracy to parameter reduction. LIAP maintains nearly 100% of baseline accuracy with a 57% reduction in parameters, surpassing all RED++ results. For a 28.5% parameter reduction, LIAP even preserves approximately 100.5% of the baseline accuracy.

ResNet-56 on CIFAR-10: Similar to the ResNet-50 results, LIAP improves accuracy over the baseline (94.09% vs. 93.26%) while reducing FLOPs and parameters by approximately 16.8% and 17%, respectively. Compared with HRank, FilterSketch, and DECORE—which achieve slightly higher compression—LIAP consistently attains better accuracy (93.84% vs. 93.52%, 93.65%, and 93.34%, respectively). Under higher pruning ratios, LIAP further decreases model complexity relative to APIB (84% vs. 83% parameter reduction and 84% vs. 81% FLOPs) while maintaining comparable accuracy.

ResNet-110 on CIFAR-10: LIAP consistently outperforms existing approaches across all evaluated settings, achieving both superior accuracy and greater model compression. It attains a 59.7% reduction in parameters while improving accuracy by 0.41% compared to leading pruning methods such as HRank (Lin *et al.*, 2020) and DECORE (Alwani *et al.*, 2021). In comparison with L1 (;and Hans Peter Graf, 2017) and GAL (Lin *et al.*, 2019), LIAP achieves higher accuracy with lower computational cost and fewer parameters.

Lightweight Networks: To further assess the performance of the proposed LIAP method, we applied it to several lightweight architectures. Table 4.6 presents the results, showing the

performance of LIAP across these networks. The final column reports the runtime (RT) of the pruning process for each network as obtained by LIAP.

Network	Dataset	Pruning Ratio	ACC (%)	RT (%)
Tiny-YOLO	CIFAR-10	25%	0.42% (↓)	0.06s
VGG-11	MNIST	50%	0.12% (↑)	0.25s
AlexNet	CIFAR-10	50%	5.00% (↑)	0.05s
Mobilenet-v2	CIFAR-10	40%	1.18 % (↓)	0.02s

Table 4.6 Accuracy Comparison on Lightweight Networks

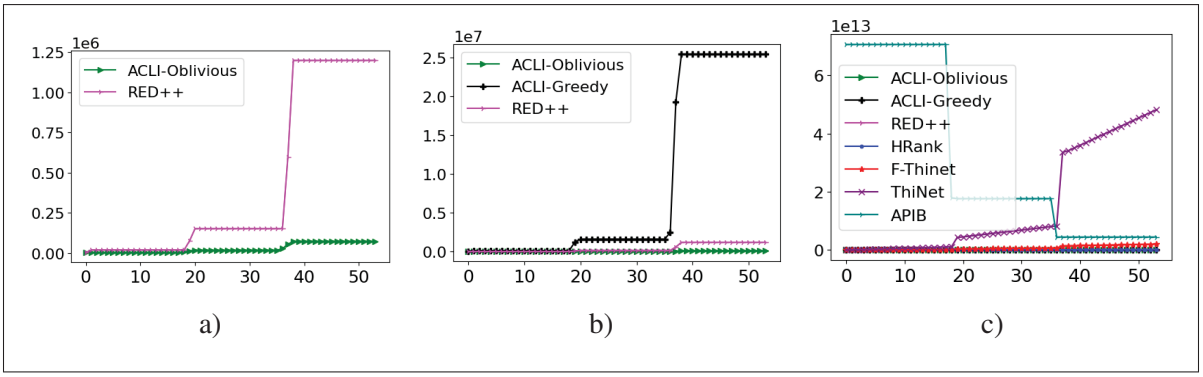

Figure 4.3 Comparison of computational complexity among pruning methods on ResNet-56 with CIFAR-10. The X-axis denotes the layer index, and the Y-axis indicates the number of multiplications involved in filter selection

Table 4.7 Comparison of the ratios of multiplications required by different pruning methods on ResNet-56 for a 50% pruning ratio

	LIAP (Oblivious)	LIAP (Greedy)	F-ThiNet	ThiNet	HRank	APIB	RED++
LIAP(Obl)	1	3.62e-03	4.25e-08	2.04e-09	2.01e-06	9.94e-10	7.28e-02
LIAP(Gr)	2.76e+02	1	1.17e-05	5.62e-07	5.56e-04	2.75e-07	2.01e+01
F-ThiNet	2.35e+07	8.53e+04	1	4.79e-02	4.74e+01	2.34e-02	1.71e+06
ThiNet	4.91e+08	1.78e+06	2.09e+01	1	9.88e+02	4.88e-01	3.58e+07
HRank	4.97e+05	1.80e+03	2.11e-02	1.01e-03	1	4.94e-04	3.62e+04
APIB	1.01e+09	3.64e+06	4.27e+01	2.05e+00	2.02e+03	1	7.32e+07
RED++	1.37e+01	4.97e-02	5.83e-07	2.80e-08	2.76e-05	1.37e-08	1

Table 4.8 Average RE Comparison Across ResNet-50, ResNet-56 and ResNet-110 Networks

	ResNet-50	**ResNet-56**	**ResNet-110**
LIAP	4.2e-07	5.6e-05	2.77e-05
NORTON (Zniyed *et al.*, 2025)	1.30e-08	4.77e-06	2.33e-06
RED++ (Yvinec *et al.*, 2023)	2.1e-09	4.1e-06	2.0e-06
HRank (Lin *et al.*, 2020)	9.3e-13	1.1e-10	2.77e-11
IPBM (Guo *et al.*, 2023)	1.1e-16	5.6e-14	2.8e-14
ThiNet	4.8e-18	-	-

4.6.2 Resource Efficiency

Maintaining the accuracy of a pruned network is essential; however, a pruning method must also be efficient and suitable for deployment on resource-limited devices. To this end, we introduce a metric termed *resource efficiency* (RE), which jointly accounts for model accuracy and pruning complexity. It is defined as:

$$RE = \left(\frac{\text{ACC}}{\text{MAC}}\right), \tag{4.29}$$

where ACC denotes the accuracy of the pruned model, and MAC represents the number of multiply–accumulate operations performed during pruning. This metric provides a balanced measure of efficiency by quantifying the trade-off between accuracy preservation and computational cost.

Table 4.8 summarizes the RE of LIAP and key baselines, HRank (Lin *et al.*, 2020), ThiNet (Luo *et al.*, 2018), NORTON (Zniyed *et al.*, 2025), and IPBM (Guo *et al.*, 2023), evaluated on ResNet-50 (ImageNet) and ResNet-56/110 (CIFAR-10). Reported accuracies are taken from the corresponding publications. As pruning ratios are not explicitly provided, they are approximated using parameter reduction. ThiNet results are available only for ResNet-50, while RED++ provides graphical results from which accuracy preservation ($\approx 95.5\%$) and parameter reductions are estimated. For RED++, the reported MACs reflect only redundancy evaluation during similarity-based clustering; thus, the actual cost is likely higher. Across pruning ratios, RE remains stable. For example, LIAP maintains an RE of about 4.2×10^{-7} on ResNet-50 for ratios

between 0.2 and 0.72. This trend is consistently observed across all networks and methods, confirming that RE is a reliable metric for evaluating the trade-off between computational cost and accuracy. Accordingly, Table 4.8 reports the average RE over these ratios.

Among the evaluated approaches, NORTON and RED++ achieve performance closest to LIAP, though their RE values remain approximately one order of magnitude lower. In contrast, IPBM and HRank exhibit substantially smaller RE values, with LIAP outperforming them by factors of about 10^9 and 10^5, respectively. These results demonstrate the effectiveness of LIAP in balancing accuracy and computational efficiency, underscoring its suitability for resource-constrained environments.

4.6.3 Complexity Comparison

In this section, we compare the computational complexity of different pruning methods for filter selection in the convolutional layers of ResNet-56 on CIFAR-10, with the results presented in Figure 4.3. Computational complexity is quantified by the number of multiplications performed during filter selection. All experiments are conducted with a pruning ratio of 50% and a batch size of 256. For ThiNet and F-ThiNet, the parameter $\mathbb{I}$, representing the number of entries in the output of the $(l+1)^{\text{st}}$ layer, is set to 10.

We also compare the relative complexity ratios across pruning methods, as shown in Table 4.7, where the element in the i^{th} row and j^{th} column is computed as $\frac{N_R(i)}{N_C(j)}$. Here, $N_R(i)$ and $N_C(j)$ denote the total number of multiplications required for filter selection by the methods in the i^{th} row and j^{th} column, respectively. The runtime of the LIAP pruning process for several lightweight CNNs is reported in Table 4.6. Additionally, the runtime of a method can be approximated using the equation proposed in (Lannelongue, Grealey & Inouye, 2021):

$$\text{Runtime} = \frac{\text{FLOPs}}{\text{FLOP/s}}. \tag{4.30}$$

Based on this equation, pruning 50% of ResNet-56 on hardware with 13.4×10^{12} FLOP/s yields an estimated runtime of 1.2×10^{-7} seconds for our method, compared to 120 seconds for APIB. Assuming comparable hardware across methods, FLOPs serve as a reliable proxy for runtime. Figure 4.3 highlights the substantial complexity gap between our oblivious algorithm and existing pruning approaches. As shown in Figure 4.3a, LIAP requires approximately 1.7×10^6 multiplications, while NORTON and RED++ demand 2.0×10^7 and 2.3×10^7, making them roughly 11.8 and 13.5 times more computationally intensive, respectively. Figure 4.3b compares the oblivious and greedy versions of LIAP, alongside NORTON and RED++. The greedy algorithm exhibits substantially higher layer-wise complexity, requiring about 2.76×10^2 times more multiplications overall, effectively flattening the oblivious curve in the plot. According to Figure 4.3c, APIB incurs the highest computational cost among all methods, except ThiNet in deeper layers, requiring approximately 1.01×10^9 times more multiplications than LIAP.

As shown in the first row of Table 4.7, the total number of multiplications required by the oblivious algorithm is 3.62×10^{-3} times that of the greedy algorithm. The corresponding ratios for F-ThiNet, ThiNet, HRank, and APIB are 4.25×10^{-8}, 2.04×10^{-9}, 2.01×10^{-6}, and 9.09×10^{-10}, respectively. Conversely, the entry in the sixth row and first column of Table 4.7 indicates that APIB requires approximately 1.0×10^9 times more multiplications than the oblivious algorithm, underscoring the substantial gap in computational demand. The ratios of multiplications for the greedy, F-ThiNet, ThiNet, and HRank methods relative to the oblivious approach are 2.76×10^2, 2.35×10^7, 4.91×10^8, and 4.97×10^5, respectively.

4.6.4 Nonzero Filter Replacement

In this section, we evaluate the efficiency of the nonzero filter replacement (FR). Designed for on-device pruning of CNNs in a simple and low-complexity manner, the FR method is assessed by comparing the accuracy reduction of a modified version of VGG-16 on the CIFAR-10 dataset. This version of VGG-16 is a slightly modified architecture tailored for 32×32 input images, such as those in the CIFAR-10 dataset. It includes 7 convolutional layers with increasing channel depth (from 64 to 256), each using 3×3 kernels, interleaved with 3 max-pooling

layers that progressively reduce the spatial resolution. After feature extraction, the output is flattened and passed through three fully connected (FC) layers, making the model suitable for a 10-class classification task. This network, designed and trained from scratch, consists of seven convolutional layers followed by three fully connected (FC) layers. It incorporates three max-pooling layers located after the second, fourth, and seventh convolutional layers. In this experiment, the FR method is compared with the random pruning method, the L1 method (;and Hans Peter Graf, 2017), and the ThiNet method (Luo *et al.*, 2018). The network was trained with a batch size of 128, using the ReLU activation function in all layers except the last FC layer, which employs the sigmoid activation function. To mitigate overfitting, logistic regression and dropout (with a dropout rate of 50%) were applied. After 100 training epochs, the network achieved approximately 81% test accuracy. This model was subsequently used in our experiments to evaluate the efficiency of various model compression methods.

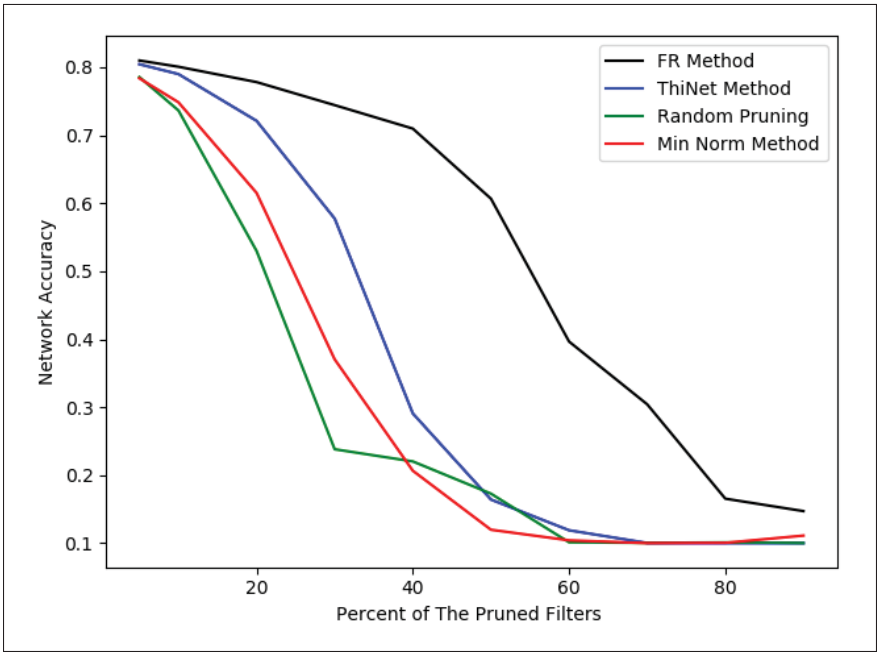

Figure 4.4 Comparison of various pruning methods with the proposed FR method for the modified VGG-16 model on the CIFAR-10 dataset

Figure 4.4 and Table 4.9 present a comparison of accuracy reduction in the pruned VGG-16 model using the proposed FR method, L1 pruning (;and Hans Peter Graf, 2017), ThiNet-50

Table 4.9 Comparison of results for the modified VGG-16 model on the CIFAR-10 dataset. Each column shows the reduction in accuracy for a given percentage of replacement using our proposed methods, while reductions for other methods are obtained through pruning

Method	**10%**	**20%**	**40%**	**60%**	**70%**
FR Method	1.16	3.42	10.26	41.60	64.70
L1 (;and Hans Peter Graf, 2017)	6.39	19.72	60.60	70.83	71.20
Random Pruning	7.59	28.26	59.22	71.12	71.14
ThiNet-50 (Luo *et al.*, 2018)	2.24	9.12	52.16	69.35	71.24

(Luo *et al.*, 2018), and random pruning. As shown in Figure 4.4, the accuracy reduction with the proposed FR method is significantly lower compared to the other methods, highlighting its efficiency for on-device model compression. Furthermore, the numerical results in Table 4.9 indicate a notable performance gap between the FR method and the other approaches, particularly at the 40% pruning level, where the FR method demonstrates superior retention of network accuracy.

4.7 Conclusion

Our proposed filter pruning method that replaces filters with zero filters, and our model compression approach that approximates the optimal nonzero filter for replacement, jointly enhance the flexibility and effectiveness of the compression process. Our proposed importance functions for filter selection exhibit weak submodularity, which guarantees the convergence of our fast, easy-to-implement, and low-complexity oblivious algorithm. Extensive experiments demonstrate that our method achieves state-of-the-art performance across various benchmarks. Furthermore, we introduced a novel Resource Efficiency (RE) metric to evaluate pruning methods and showed that our proposed approach significantly outperforms existing techniques in resource efficiency.

4.8 Supplementary Materials

4.8.1 Example

For instance, in the first convolutional layer of the VGG-16 architecture, there are 64 filters of size $3 \times 3 \times 3$, corresponding to 64 vectors in the 27-dimensional vector space $\mathbf{T}_{3\times3\times3}$. A subset $\mathcal{S}$ containing 27 linearly independent filters forms a basis for $\mathbf{T}_{3\times3\times3}$. These 27 filters can generate the remaining 37 filters, enabling all 64 output channels to be expressed as linear combinations of the output channels produced by $\mathcal{S}$. In this scenario, computing a single output entry through convolution requires 27 multiplications, identical to the number needed for the linear combination method. To achieve computational efficiency, the size of the generating set $\mathcal{S}$ must be reduced to fewer than 27.

4.8.2 Proof of Theorem 6

In this paper, we use the regular definition of convolution operation in a convolutional layer. The $[m, n, s]$ entry in the output of l^{th} convolutional layer is obtained by convolving the s^{th} filter, F^l_{sF}, by part of the input, $\bigoplus_l \overleftrightarrow{(m, n, s)}$. The convolution operation is the inner product of F^l_{sF} and $\bigoplus_l \overleftrightarrow{(m, n, s)}$. In order to proof the theorem, we calculate the absolute error in the output of the $(l + 1)^{st}$ layer. The $[m, n, s]$ entry in the output of the $(l + 1)^{st}$ layer is as follows

$$\begin{aligned} X_{l+1}[m, n, s] &= \left\langle F^{l+1}_{\{s\}F}, \text{ReLU}\left(\bigoplus_l \overleftrightarrow{(m, n, s)}\right)\right\rangle \\ &= \sum_{p=1}^{H_l} \sum_{i,j=1}^{k_{l+1}} f^{l+1}_{(i,j,p,s)} \text{ReLU}\left(\mathcal{A}^{m,n,s}_{i,j,p}\right), \end{aligned} \tag{4.31}$$

where

$$\begin{aligned}\mathcal{A}^{m,n,s}_{i,j,p} &= \left(\bigoplus_{l} \overleftrightarrow{(m,n,s)}\right)[i,j,p] \\ &= \left\langle F^{l}_{pF}, ReLU\left(\bigoplus_{l-1}\left(\overleftrightarrow{\bigoplus_{l}(m,n,s)}[i,j,p]\right)\right)\right\rangle .\end{aligned} \tag{4.32}$$

Similarly, $\hat{X}^{p_r}_{l+1}[m,n,s]$, the $[m,n,s]$ entry of the X_{l+1} when the p^{th}_r filter from the l^{th} layer, $F^{l}_{\{p_r\}F}$is replaced with $\hat{F}^{l}_{\{p_r\}F}$, is as follows

$$\begin{aligned}\hat{X}^{p_r}_{l+1}[m,n,s] = &\sum_{p=1}^{p_r-1}\sum_{i,j=1}^{k_{l+1}} f^{l+1}_{(i,j,p,s)}\, ReLU(\mathcal{A}^{m,n,s}_{i,j,p}) \\ &+ \sum_{p=p_r+1}^{H_l}\sum_{i,j=1}^{k_{l+1}} f^{l+1}_{(i,j,p,s)}\, ReLU(\mathcal{A}^{m,n,s}_{i,j,p}) \\ &+ \sum_{i,j=1}^{k_{l+1}} \hat{f}^{l+1}_{(i,j,p_r,s)}\, ReLU(\mathcal{A}^{m,n,s}_{i,j,p_r}).\end{aligned} \tag{4.33}$$

Then the absolute error in the the $[m,n,s]$ entry of X_{l+1} before and after pruning is calculated as follows

$$\begin{aligned}&|X_{l+1}[m,n,s] - \hat{X}^{p_r}_{l+1}[m,n,s]|^2 \\ &= \sum_{i,j=1}^{k_{l+1}} \left(f^{l+1}_{(i,j,p_r,s)}\, ReLU(\mathcal{A}^{m,n,s}_{i,j,p})\right)^2 \\ &\leq \left(\sum_{i,j=1}^{k_{l+1}} \left(f^{l+1}_{(i,j,p_r,s)}\right)^2\right)\left(\sum_{i,j=1}^{k_{l+1}} \left[(\mathcal{A}^{m,n,s}_{i,j,p})^2 - (\mathcal{A}^{m,n,s}_{i,j,p_r})^2\right]\right) \qquad (4.34) \\ &\leq \left\|F^{l+1}_{\{p_r\}C}\right\|_2^2 \left\|F^{l}_{\{p_r\}F} - \hat{F}^{l}_{\{p_r\}F}\right\|_2^2 \alpha^{l-1}_{(m,n)}. \qquad (4.35)\end{aligned}$$

where

$$\alpha_{(m,n)}^{l-1} = \sum_{i,j=1}^{k_{l+1}} \left(\| \, Relu \left(\bigoplus_{l-1} \left(\overleftrightarrow{\bigoplus_{l} (m,n)[i,j,p_r]} \right) \right) \|_2^2 \right). \tag{4.36}$$

Equations (4.34) and (4.35) can be hold from Cauchy-Schwarz inequality and lemma (1) proven in (Srinivas & Babu, 2015).

Therefore, the inequality in Theorem 1 holds by summing over the values of m, n, and s as follows

$$\begin{aligned} \|X_{l+1} - \hat{X}_{l+1}^{p_r}\|_2^2 &= \sum_{s,m,n} |X_{l+1}[m,n,s] - \hat{X}_{l+1}^{p_r}[m,n,s]|^2 \\ &\leq \sum_{s,m,n} \|F_{(:,:,:,p_r,s)}^{l+1}\|_2^2 \|F_{(:,:,:,p_r)}^{l}\|_2^2 \alpha_{(m,n)}^{l-1} \\ &= \|F_{(:,:,:,p_r,:)}^{l+1}\|_2^2 \|F_{(:,:,:,p_r)}^{l}\|_2^2 \Lambda_l \end{aligned} \tag{4.37}$$

where

$$\Lambda_l = \sum_{m=1}^{W_{l+1}} \sum_{n=1}^{L_{l+1}} \alpha_{(m,n)}^{l-1}. \tag{4.38}$$

In this Theorem,

$$\Lambda_1 = \sum_{m=1}^{W_2} \sum_{n=1}^{L_2} \alpha_{(m,n)}^{0},$$

where

$$\alpha_{(m,n)}^{0} = \sum_{i,j=1}^{k_2} \left(\| \, Relu \left(\bigoplus_{0} \left(\overleftrightarrow{\bigoplus_{1} (m,n)[i,j,p_r]} \right) \right) \|_2^2 \right).$$

which $\bigoplus_0$ represent a part of a input image to the network.

4.8.3 Proof of Theorem 8

Given set $V = \{1, 2, \ldots, H_l\}$, the set of indices of the filters in l^{th} convolutional layer, for every $U \subset V$, $L \subset U$, and $i \notin L$

$$\mathcal{L}_{pr}(i|L) = \max\{\| F^l_{(L\cup\{i\})F} \|_2^2, \| F^{l+1}_{(L\cup\{i\})C} \|_2^2\} \tag{4.39}$$

$$- \max\{\| F^l_{LF} \|_2^2, \| F^{l+1}_{LC} \|_2^2\}$$

$$= \max\{\| F^l_{LF} \|_2^2 + \| F^l_{\{i\}F} \|_2^2 \tag{4.40}$$

$$, \| F^{l+1}_{LC} \|_2^2 + \| F^{l+1}_{\{i\}C} \|_2^2\} \tag{4.41}$$

$$- \max\{\| F^l_{LF} \|_2^2, \| F^{l+1}_{LC} \|_2^2\} > 0. \tag{4.42}$$

Now, let's assume that $R \subset V$ is a set that is disjoint from the set U such that $|R| \leq k$. Therefore, $\sum_{i\in R} \mathcal{L}_{pr}(i|L) > 0$. Setting $\gamma_{U,k}$ as follows

$$\gamma_{U,k} = \min_{L\subset U} \min_{R\subset V, |R|\leq k} \frac{\sum_{i\in R} \mathcal{L}_{pr}(i|L)}{\mathcal{L}_{pr}(R|L)}, \tag{4.43}$$

the inequality condition in Equation (4.14) is satisfied. Because U has finite subsets, and for every set $L \subset U$ there is a finite number of such sets R, so, the feasible set in Equation (4.43) is finite. As a result, a positive number $\gamma_{U,k}$ exists. Since for every set $U \subset \{1, 2, \ldots, H_l\}$ and every integer number $k \in \mathbb{N}^+$, the submodularity ratio $\gamma_{U,k}$ exists, the set function $\mathcal{L}_{pr}$ is a $\gamma_{U,k}$-weakly submodular.

As follows, we explore the submodularity of $\| \cdot \|_2^2$ as a set function on the set of filters in a convolutional layer. For the set of filters in the l^{th} layer of a CNN, the function $\| \cdot \|_2^2$ can be regarded as a set function. Given a subset S of $\{1, 2, \ldots, H_l\}$, the index set of the filters in the l^{th} layer, it returns $\| F^l[:, :, :, S] \|_2^2$. The following lemma can be easily proven for this set function.

Lemma 3. *For any subset $S \subset \{1, 2, \ldots, H_l\}$ and any index $\alpha \notin S$ we have:*

$$\| F^l[:, :, :, S \cup \{\alpha\}] \|_2^2 - \| F^l[:, :, :, S] \|_2^2 = \| F^l[:, :, :, \alpha] \|_2^2, \tag{4.44}$$

and

$$\| F^{l+1}[:,:,S \cup \{\alpha\},:] \|_2^2 - \| F^{l+1}[:,:,S,:] \|_2^2 = \| F^{l+1}[:,:,\alpha,:] \|_2^2 . \tag{4.45}$$

The following theorem shows that $\| . \|_2^2$ is a submodular set function.

Theorem 11. *The submodularity ratio of the set function $\| . \|_2^2$ is 1. The set of size K selected by the oblivious algorithm equipped with the $\| . \|_2^2$, as the importance function, is the optimal solution.*

Proof: Let's assume $U \subset V$. For any $L \subset U$ and $S \subset V$ where $|S| \leq k$

$$\gamma_{U,k} = \min_{L \subset U} \min_{S \subset V, |S| \leq k} \left(\frac{\sum_{i \in S} \| i|L \|_2^2}{\| S|L \|_2^2} \right) \tag{4.46}$$

$$= \min_{L \subset U} \min_{S \subset V, |S| \leq k} \left(\frac{\sum_{i \in S} \| F^l_{(L \cup \{i\})F} \|_2^2 - \| F^l_{LF} \|_2^2}{\| F^l_{(L \cup \{S\})F} \|^2 - \| F^l_{LF} \|^2} \right) \tag{4.47}$$

$$= \min_{L \subset U} \min_{S \subset V, |S| \leq k} \left(\frac{\sum_{i \in S} \| F^l_{\{i\}F} \|_2^2}{\| F^l_{SF} \|_2^2} \right) = 1. \tag{4.48}$$

This theorem shows that when the filter selection criteria is only the l_2-norm of the filters, the $\gamma_{U,k} = 1$. It means that the $\| . \|_2^2$ is a submodular set function . On the other hand, since

$$\sum_{i \in S} \| F^l_{\{i\}F} \|_2^2 = \| F^l_{SF} \|_2^2, \tag{4.49}$$

a set S of filters has the maximum l_2-norm when the filters in S individually have the greatest norms among the filters in the l^{th} layer. Therefore, the oblivious algorithm equipped with $\| \ldots \|_2^2$ provides the optimal solution to the maximization problem (1) in the paper.

Remark 5. *In a convolutional layer, if $\| F^l_{\{i\}F} \|_2^2 \geq \| F^{l+1}_{\{i\}C} \|_2^2$ for all $i = 1, \ldots, H_l$, then the importance function $\mathcal{L}_{pr}$ is submodular, and $S^{obv} = S^*$.*

Remark 6. *In a convolutional layer, if* $\| F^l_{\{i\}F} \|_2^2 \leq \| F^{l+1}_{\{i\}C} \|_2^2$ *for all* $i = 1, \ldots, H_l$*, then the importance function* $\mathcal{L}_{pr}$ *is submodular, and* $S^{obv} = S^*$.

4.8.4 Proof of Theorem 9

Let S^{obv} denote the set selected by the oblivious algorithm, and let S^* represent the optimal index set of k filters. By definition of the oblivious algorithm ($\sum_{i \in S^{obv}} \mathcal{L}_{pr}(i) \geq \sum_{i \in S^*} \mathcal{L}_{pr}(i)$), using Equations (4.6) and (4.17), the following inequalities are hold

$$\mathcal{L}_{pr}(S^{obv}) \geq \frac{\sum_{i \in S^{obv}} \mathcal{L}_{pr}(i)}{k} \geq \frac{\sum_{i \in S^*} \mathcal{L}_{pr}(i)}{k} \geq \frac{\gamma_{\emptyset,k}}{k} \mathcal{L}_{pr}(S^*). \tag{4.50}$$

which proves Equation (4.19). To prove Equation (4.20), without loss of generality, we assume $\mathcal{L}_{pr}(S^{obv}) = \| F^l_{S^{obv}F} \|_2^2$, therefore

$$\begin{aligned}
\mathcal{L}_{pr}(S^{obv}) &= \| F^l_{S^{obv}F} \|_2^2 && (4.51)\\
&= \| F^l_{S_1^{obv}F} \|_2^2 + \| F^l_{S_2^{obv}F} \|_2^2 && (4.52)\\
&\leq \| F^l_{S_1^{obv}F} \|_2^2 + \| F^{l+1}_{S_2^{obv}C} \|_2^2 . && (4.53)
\end{aligned}$$

Since $\mathcal{L}_{pr}(S^{obv}) = \| F^l_{S^{obv}F} \|_2^2$, consequently

$$\begin{aligned}
&\| F^l_{S_1^{obv}F} \|_2^2 + \| F^l_{S_2^{obv}F} \|_2^2 \geq \| F^{l+1}_{S_1^{obv}C} \|_2^2 + \| F^{l+1}_{S_2^{obv}C} \|_2^2 \\
\Rightarrow &2 \| F^l_{S_1^{obv}F} \|_2^2 + \| F^l_{S_2^{obv}F} \|_2^2 - \| F^{l+1}_{S_1^{obv}C} \|_2^2 \\
&\geq \| F^l_{S_1^{obv}F} \|_2^2 + \| F^{l+1}_{S_2^{obv}C} \|_2^2 && (4.54)\\
\Rightarrow &\mathcal{L}_{pr}(S^{obv}) + (\| F^l_{S_1^{obv}F} \|_2^2 - \| F^{l+1}_{S_1^{obv}C} \|_2^2) \\
&\geq \sum_{i \in S^{obv}} \mathcal{L}_{pr}(i) \geq \sum_{i \in S^*} \mathcal{L}_{pr}(i) \geq \gamma_{\emptyset,k} \mathcal{L}_{pr}(S^*), && (4.55)
\end{aligned}$$

now by defining $\mathcal{R}_{obv} = \| F^l_{S_1^{obv}F} \|_2^2 - \| F^{l+1}_{S_1^{obv}C} \|_2^2$, Equation (4.20) is proven. Similarly by assuming $\mathcal{L}_{pr}(S^{obv}) = \| F^{l+1}_{S^{obv}C} \|_2^2$ the inequality hold for $\mathcal{R}_{obv} = \| F^{l+1}_{S_2^{obv}C} \|_2^2 - \| F^l_{S_2^{obv}F} \|_2^2$.

CHAPTER 5

A LOW-COMPLEXITY MODIFIED THINET ALGORITHM FOR PRUNING CONVOLUTIONAL NEURAL NETWORKS

Sadegh Tofigh[1] , M. Omair Ahmad [2] , M. N. S. Swamy [2]

[1] Department of Electrical Engineering, École de Technologie Supérieur,
1100 Notre-Dame Ouest, Montréal, Québec, Canada H3C 1K3

[2] Department Electrical and Computer Engineering, Concordia University,
1455 Blvd. De Maisonneuve Ouest, Montreal, Quebec, Canada H3G 1M8

Article accepted by IEEE Signal Processing Letters, April 2022.

Abstract

ThiNet is a recent method for pruning convolutional neural networks. This method uses a norm of a subset of the components of the output resulting from the convolutional layer succeeding the layer from which the filters are to be removed for pruning the network. The ThiNet algorithm is very time-consuming, in view of the fact that the filters for removal are selected one by one iteratively. In this paper, we propose a modified version of ThiNet, in which the same information on the output of the same convolutional layer as used by ThiNet is employed to select all the filters together in a single step, for pruning the network. The proposed modified algorithm is shown to have a time-complexity that is only a small fraction of that of ThiNet or any other state-of-the-art algorithm, and the pruned network has almost the same reduction in its accuracy as that of the network pruned by ThiNet.

Keywords: Network pruning, Convolutional neural networks, ThiNet algorithm, CNN model compression

5.1 Introduction

Inspired by the research in the area of the human visual cortex, the idea of convolutional layers was first introduced in 1980 by Fukushima (Fukushima, 1980). However, an architecture of a convolutional neural network was initially proposed in 1989 by Lecun et al. (LeCun *et al.*,

1989) for the task of handwritten digit recognition. However, it is only after powerful graphical processing units became available that a large variety of deep convolutional networks have been proposed for their use in diverse applications (Goodfellow *et al.*, 2014), (Hong *et al.*, 2015), (Zhou, Wei, Zhu & Collin, 2021). Even though deep CNNs have proved to be very useful in many applications, applications involving portable and mobile devices that require low-power consumption and have limited storage capacity cannot benefit from the advantages provided by these deep networks.

Filter pruning is a model compression approach that aims to compress and simplify a high-complexity deep CNN by removing some of the filters from its convolutional layers, such that their removal reduces the network complexity without a significant reduction in the accuracy (Cheng *et al.*, 2017). Recently, several algorithms have been proposed for pruning convolutional neural networks(;and Hans Peter Graf, 2017), (He *et al.*, 2019b), (Luo *et al.*, 2018), (Tian, Chen, Zeng & Liu, 2021), (Fan, Su, Jing & Lu, 2021), (Ding, Ding, Han & Tang, 2018), (Luo & Wu, 2020), (Lin *et al.*, 2020), (Lin *et al.*, 2019). Each of these algorithms gives a different mechanism for selecting the filters to be removed from the convolutional layers. The authors of (Luo *et al.*, 2018) have proposed a pruning algorithm called ThiNet algorithm, whose computational complexity is very high. In (Tian *et al.*, 2021), (Fan *et al.*, 2021), (Ding *et al.*, 2018), (Luo & Wu, 2020), (Lin *et al.*, 2020), (Lin *et al.*, 2019), authors have attempted to improve slightly the classification accuracy of the networks pruned by developing their own pruning algorithms; however, even though the computational complexities of their algorithms are somewhat lower than that of ThiNet, they are still quite high.

In this paper, we propose a faster version of the ThiNet algorithm (F-ThiNet) in which all the filters to be removed from the network can be selected in a single step, rather than selecting the filters one by one iteratively. The performance of the proposed algorithm, in terms of the accuracy reduction of the pruned network, is studied on two networks, including the VGG-16 network, and it is shown that the accuracy reduction of the pruned network is almost the same as that of the network pruned by the ThiNet algorithm.

Notations: In the following study, a tensor $F^l_{(:,:,:,:)} = [f^l_{i,j,p,q}]$ is a 4-D tensor representing

the filter set in the l^{th} layer ($l = 1, ..., \Omega$), $f^l_{i,j,p,q}$ represents the entry located in the i^{th} row, j^{th} column and the p^{th} channel of the q^{th} filter in the convolutional layer l ($i = 1, ..., k_l; j = 1, ..., k_l; p = 1, ..., H_{l-1}; q = 1, ..., H_l$). $F^l_{(:,:,:,q)}$ represents the q^{th} filter in the l^{th} layer. X^l of size $L_l \times W_l \times H_l$ represents the output of the layer l, where L_l and W_l are, respectively, the height and width of X^l. $X^l[m, n, s]$ represents the entry in the m^{th} row, n^{th} column and s^{th} channel of X^l.

5.2 F-ThiNet Algorithm

In the ThiNet method, K filters from a total of H_l filters of the l^{th} convolutional layer are removed so that their removal has the minimum effect on an aggregate value of a combination of the values of some selected entries in the output of the $(l + 1)^{st}$ convolutional layer, using m input images. Determining which set of K filters out of H_l filters is to be selected for pruning is an NP-hard problem. In order to overcome this problem, the ThiNet method, in the framework of a greedy algorithm, uses the equations

$$A_T = \sum_{i=1}^{m} \left(\sum_{j \in T} \hat{X}^{l+1}_{i,j}[p_1, p_2, p_3] \right)^2 \tag{5.1}$$

and

$$\arg\min_T \{A_T\} \tag{5.2}$$

where $\hat{X}^{l+1}_{i,j}[p_1, p_2, p_3]$ is the value of the entry of the output of the $(l + 1)^{st}$ convolutional layer generated by the i^{th} input image and filter j in the subset T of indices of the filters in the l^{th} convolutional layer. In this algorithm, the K filters are selected using the procedure described below.

To start with, the set T is empty. In each iteration, the index of one additional filter from the l^{th} convolutional layer is added to it. At the beginning of the r^{th} iteration, the set T has indices of $r - 1$ that were selected from the l^{th} convolutional layer in the previous iterations. During the r^{th} iteration, the size of the set T is increased by unity by adding to it the index of that filter from the remaining filters of the l^{th} convolutional layer that uses (5.1) and (5.2). When the filter in the

Algorithm 5.1 Proposed F-ThiNet algorithm

Input: Training set $\{(\hat{x}_i, \hat{y}_i)\}$, and the compression rate R
Output: The subset of removed channels: T

$T \leftarrow \emptyset$;
$I \leftarrow \{1, 2, \ldots, H_l\}$;
$VALUE \leftarrow []$;
$K \leftarrow H_l \times R$;
for *each item* $i \in I$ **do**
    $tmpT \leftarrow T \cup \{i\}$;
    Calculate $value$ from Eq. (5.1) using $tmpT$;
    Add $value$ to $VALUE$;
end for
$MAXIMUM \leftarrow \max(VALUE)$;
while $|T| \leq K$ **do**
    $Minind \leftarrow \text{argmin}(VALUE)$;
    Add $Minind$ to T;
    $VALUE[Minind] \leftarrow MAXIMUM + 1$;
end while

r^{th} iteration is selected, the argmin in (5.2) operates on $H_l - r + 1$ values of A_T computed using (5.1). For example, for the selection of the first filter from the l^{th} convolutional layer, the argmin in (5.2) operates on H_l values computed in (5.1). For each of these H_l values, the number of multiplications per image needed is given by

$$N_{l,r,I} = \left[\left(\sum_{i=1}^{l-1} C_{i,H_i} \right) + C_{l,r} + rIk_{l+1}^2 \right] \tag{5.3}$$

where $C_{\mu,\nu} = \nu k_\mu^2 H_{\mu-1} L_\mu W_\mu$ $(\mu = 1, ..., \Omega; \nu = 1, ..., H_\mu)$ and I is the number of selected entries in the output of the $(l+1)^{st}$ convolutional layer. The three terms in this equation compute, respectively, the number of multiplications in the convolutional layers preceding the pruned layer l , the number of the multiplications in the l^{th} layer when it contains r filters, and the number of multiplications in the $(l+1)^{st}$ convolutional layer when each filter has r channels.

In this section, we now present a fast version of ThiNet algorithm as Algorithm 1 and refer to it as F-ThiNet algorithm. In this algorithm, for each of the filters $F^{l}_{(:,:,:,q)}$ $(q = 1, ..., H_l)$ in the l^{th} convolutional layer, $A_{\{q\}}$ is computed using (5.1) Then, the set of K indices, $i_1, \cdots, i_K$, is chosen from the indices of all the filters in the l^{th} convolutional layer for which the corresponding $A_{\{q\}}$ values are the lowest and these K corresponding filters are selected for pruning.

It needs to be noted that in order to select K filters from the l^{th} convolutional layer using the ThiNet algorithm, the number of multiplications needed is $\sum_{i=1}^{K}(H_l - i + 1)N_{l,i,I}$. On the other hand, the number of multiplications needed to select K filters from the l^{th} convolutional layer using the F-ThiNet algorithm is only $H_l N_{l,1,I}$, which is independent of K. Therefore, the computations needed to select K filters in the l^{th} convolutional layer in the F-ThiNet algorithm is the same as the amount of computations needed for the selection of the first filter in the ThiNet algorithm, thus making the proposed F-ThiNet algorithm to have a significantly reduced complexity.

5.3 Experiments

5.3.1 Performance and Comparison with ThiNet

In order to empirically evaluate the performance of the proposed F-ThiNet algorithm and compare it with that of the ThiNet algorithm, the two algorithms are individually employed for pruning two different networks, VGG-16 and AlexNet, on two datasets, CIFAR-10 and Fashion-MNIST. The CIFAR-10 dataset includes 50,000 training and 10,000 testing RGB images, each of size $32 \times 32 \times 3$. The Fashion-MNIST dataset includes 50,000 training and 10,000 testing grey-level images, each of size 28×28. For each of the datasets, a subset of 1000 images is randomly chosen for the process of pruning when applying either of the two algorithms. Since in both the ThiNet and F-ThiNet algorithms, the accuracy reduction in the output of the network is determined by using the information from the convolutional layer that immediately follows the convolutional layer from which the filters are pruned, these two methods cannot be applied to assess the effect of pruning the filters from the last convolutional layer of a network. Hence, in

our experiments, we exclude the last convolutional layer from pruning filters from it. In all the experiments, the pruned networks are not fine-tuned.

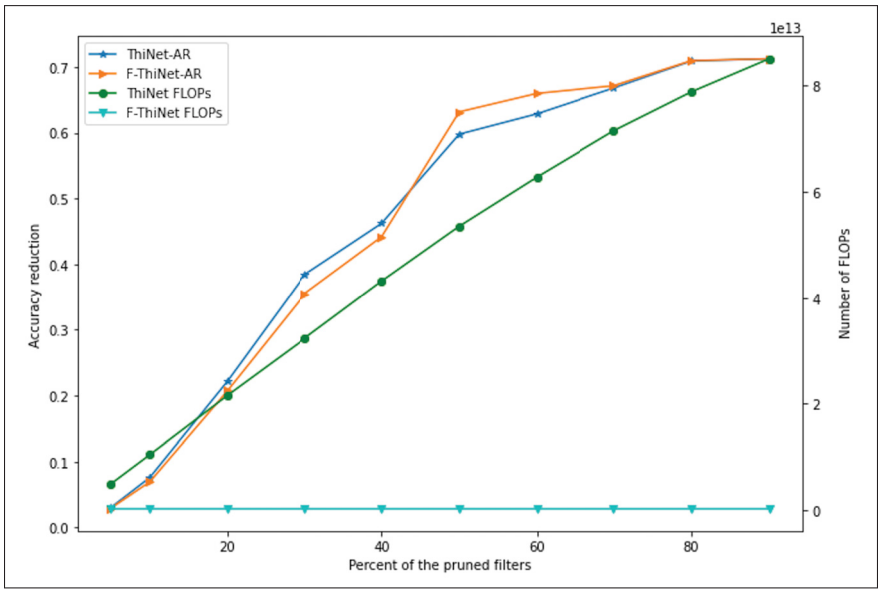

Figure 5.1 Accuracy reduction (AR) of VGG-16 with CIFAR-10 dataset images as its input, when the same % of filters from all its convolutional layers is pruned using ThiNet and F-ThiNet vs number of FLOPs required

We evaluate the performance of the two pruning algorithms in two cases of pruning. In the first case, we remove a certain percentage of filters from the first to the last but one convolutional layer, one at a time, and determine the resulting accuracy reduction of the network. In the second case, we study the effect of removing a given percentage of filters simultaneously from all the convolutional layers, excluding the last one. In both cases, we also determine the number of FLOPs required for pruning by the algorithm being assessed.

As the image sizes in the two datasets are rather small, for good performance of the networks, we do not need to use the same depth as originally employed by the two networks. Accordingly, the VGG-16 is modified to have only 7 convolutional layers followed by 3 fully connected layers and the AlexNet is modified to have 5 convolutional layers followed by 3 fully connected layers. The activation function used in each of the layers is ReLu, except for the last fully-connected

layer, in which the sigmoid function is used for activating the extracted features.

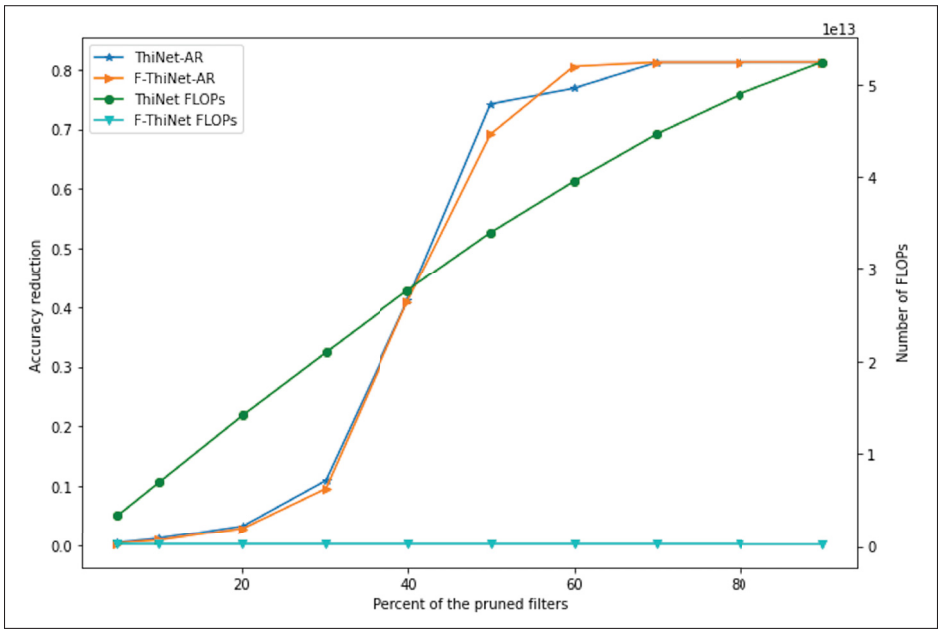

Figure 5.2 Accuracy reduction (AR) of VGG-16 with Fashion-MNIST dataset images as its input, when the same % of filters from all its convolutional layers is pruned using ThiNet and F-ThiNet vs number of FLOPs required

Since our experiments are performed on two different networks using two different datasets, the results of the performance comparison can be divided in four different parts.

First, we discuss the performance of the two algorithms when they are applied to the VGG-16 network using images from the CIFAR-10 dataset. Table 5.1 shows the accuracy reduction and the number of FLOPs needed by each of the two pruning methods when only one given convolutional layer is pruned. It is seen from this table that although the reduction in the accuracy of the network using the two algorithms is similar, F-ThiNet requires a substantially smaller number of FLOPs than that required by the ThiNet algorithm. It is noted from this table that the number of operations required by F-ThiNet is reduced by more than 98% over that required by ThiNet irrespective of the convolutional layer being pruned. Fig. 5.1 shows the accuracy reduction of the network and the number of FLOPs needed by the pruning algorithms when the network is pruned by removing the same percentage of filters from all the convolutional layers. It

is seen from this figure that generally there is no significant difference in the accuracy reduction of the pruned networks resulting from the two algorithms, irrespective of the percentage of filters removed. On the other hand, the proposed F-ThiNet algorithm requires a much smaller number of FLOPs for pruning the network. Moreover, the proposed algorithm requires the same small number of FLOPs, 4.43×10^{11}, for pruning the network irrespective of the percentage of filters pruned from the network. It needs to be pointed out that this number is the same as that required by ThiNet when only one filter is removed from each of the layers. However, the number of FLOPs required by the ThiNet algorithm rises almost linearly, but very sharply, as the percentage of the number of filters pruned increases.

The corresponding pruning results obtained by applying the two algorithms on VGG-16 using Fashion-MNIST dataset, AlexNet using CIFAR-10 dataset, and AlexNet using Fashion-MNIST dataset are depicted in Tables 5.2 -5.4, respectively. Figure 5.2 shows the accuracy reduction of VGG-16 using Fashion-MNIST dataset and the number of FLOPs needed by the pruning algorithms when the network is pruned by removing the same percentage of filters from all the convolutional layers. By examining the results provided by these tables and Figure 5.2, one can reach the same conclusions on the performance of the two algorithms as in the case of the VGG-16 network using CIFAR-10 dataset.

Table 5.1 Number of FLOPs required by ThiNet and F-ThiNet when applied to VGG-16 to prune the same % of filters from a given convolutional layer with r= 0.6 using CIFAR-10 and reduction in accuracy of the resulting pruned network

		Conv1	**Conv2**	**Conv3**	**Conv4**	**Conv5**	**Conv6**
ThiNet	**FLOPs**	3.64e+09	7.953e+10	1.055e+12	2.116e+12	1.481e+13	2.422e+13
	ACC.Red (%)	4.24	15.79	9.2	10.46	5.47	3.98
F-ThiNet	**FLOPs**	**7.168e+06**	**5.577e+08**	**1.881e+10**	**3.783e+10**	**1.509e+11**	**2.351e+11**
	ACC.Red (%)	4.17	15.21	9.76	11.12	4.68	3.89
Percentage reduction in the number of FLOPs by F-ThiNet over ThiNet		99.8	99.3	98.2	98.2	99.0	99.0

Table 5.2 Number of FLOPs required by ThiNet and F-ThiNet when applied to VGG-16 to prune the same % of filters from a given convolutional layer with r=0.7 using Fashion-MNIST and reduction in accuracy of resulting pruned network

		Conv1	Conv2	Conv3	Conv4	Conv5	Conv6
ThiNet	**FLOPs**	8.72e+08	4.417e+10	6.767e+11	1.237e+12	8.17e+12	1.189e+13
	ACC.Red (%)	1.72	2.5	0.47	1.95	0.47	0.94
F-ThiNet	**FLOPs**	**2.56e+06**	**2.464e+08**	**1.607e+10**	**2.876e+10**	**1.076e+11**	**1.497e+11**
	ACC.Red (%)	1.56	2.73	0.62	1.64	0.78	0.78
Percentage reduction in the number of FLOPs by F-ThiNet over ThiNet		99.7	99.4	97.6	97.7	98.7	98.7

Table 5.3 Number of FLOPs required by ThiNet and F-ThiNet when applied to AlexNet to prune the same % of filters from a given convolutional layer with r = 0.7 using CIFAR-10 and reduction in accuracy of resulting pruned network

		Conv1	Conv2	Conv3	Conv4
ThiNet	**FLOPs**	2.846e+09	2.248e+12	2.787e+13	7.037e+13
	ACC.Red (%)	0	3.94	2.34	2.73
F-ThiNet	**FLOPs**	**4.224e+06**	**1.281e+09**	**1.721e+11**	**5.501e+11**
	ACC.Red (%)	0	4.78	2.7	3.09
Percentage reduction in the number of FLOPs by F-ThiNet over ThiNet		99.9	99.9	99.4	99.2

Table 5.4 Number of FLOPs required by ThiNet and F-ThiNet when applied to AlexNet to prune the same % of filters from a given convolutional layer with r = 0.7 using Fashion-MNIST and reduction in accuracy of resulting pruned network

		Conv1	Conv2	Conv3	Conv4
ThiNet	**FLOPs**	2.587e+09	3.695e+12	4.116e+13	9.217e+13
	ACC.Red (%)	0.88	0.99	0.21	0.09
F-ThiNet	**FLOPs**	**3.84e+06**	**1.477e+09**	**2.854e+11**	**7.39e+11**
	ACC.Red (%)	0.74	1.01	0.19	0.14
Percentage reduction in the number of FLOPs by F-ThiNet over ThiNet		99.9	99.96	99.3	99.2

5.3.2 Comparison with other State-of-the-Art Pruning Methods

In this section, we provide a brief comparison of the performance of the proposed F-ThiNet algorithm with that of some of the other state-of-the-art pruning methods. For this purpose, we choose four recent pruning methods, namely CURL (Luo & Wu, 2020), HRank (Lin *et al.*, 2020), GAL-joint (Lin *et al.*, 2019), and FPGM (He *et al.*, 2019b), all of which have been compared with the ThiNet pruning method by applying them to prune ResNet-50. The first three of these methods make use of the ImageNet dataset (Deng *et al.*, 2009), whereas the last one uses ILSVRC-2012 dataset. The pruning method CURL outperforms the ThiNet method in improving its performance in that the reduction in the accuracy of the pruned network obtained by using CURL is 4.97% lower than that of the network pruned by using ThiNet, while the number of FLOPs in the two pruned networks is the same. On the other hand, the performance of the network pruned by the HRank method is marginally superior to that pruned by ThiNet in that the reduction in accuracy is 0.68% lower than that of the network pruned by ThiNet, while the number of FLOPs required by the former is 10.9% lower than that required by the network pruned by the latter. The GAL-joint pruning method provides an accuracy reduction of 0.89% lower than that provided by the network pruned using ThiNet, with a slight increase in the number of FLOPs required over that required by ThiNet. Finally, the FPGM pruning method lowers the accuracy reduction of the pruned network by 0.28% over that pruned by ThiNet, while the number of FLOPs required by the former is 5.5% lower than that required by the latter. From the above analysis of the results, it is seen that the reduction in the classification accuracy of the networks pruned by these four methods is lower than that of the network pruned by ThiNet, and it ranges from 0.28% to 4.97%, while the reduction in the number of FLOPs required by the pruning algorithms ranges from almost 0% to 10.9%. Hence, none of these four pruning methods has a significant advantage over that of ThiNet either in terms of the classification accuracy of the pruned networks or in the number of FLOPs required by them.

It has been seen in Section 5.3.1 that the reduction in the accuracy between the unpruned network and the network pruned by either ThiNet or F-thiNet is about the same. However, there is a very significant difference in the number of FLOPs required by the two pruning methods in that the

number FLOPs required by F-ThiNet is less than 2% of that required by ThiNet. Hence, it can be concluded that in terms of the computational complexity, our proposed pruning algorithm has a very substantial advantage over any of the pruning algorithms in the literature.

5.4 Conclusion

In this paper, a significantly faster version of the ThiNet algorithm, referred to as the F-ThiNet algorithm, has been proposed for pruning a convolutional network. It has been shown that the reduction in the accuracy of the pruned networks resulting from the two algorithms is about the same when the network is pruned by the same amount. A major advantage of the proposed algorithm is that its time-complexity is significantly lower than that of ThiNet or any other state-of-the-art algorithm. Specifically, the time-complexity of the proposed algorithm is almost two orders of magnitude lower than that of ThiNet. This low time-complexity is achieved in view of the fact that in the proposed algorithm, the selection of the filters to be removed from the layers of the network is not carried out in an iterative manner, as done by the ThiNet algorithm. Further, the time-complexity of the proposed algorithm is independent of the percentage of the filters removed from the network. This is in contrast to the ThiNet algorithm, for which the time-complexity rises almost linearly, but very sharply, as the number of filters removed increases.

CHAPTER 6

SUMMARY AND DISCUSSION

The primary objective of this thesis is to establish a mathematically grounded and sustainable framework for the structural pruning of Convolutional Neural Networks. By shifting away from empirical heuristics, the research focuses on modeling the functional inter-layer dependencies between adjacent convolutional filters to enable a completely data-free compression process. This work is structured around three core themes that bridge the gap between theoretical derivation and practical engineering. The first theme, explored in Chapter 3, presents a pruning methodology established on the derivation of an analytical upper bound for the Average Absolute Error propagated to the layer following a pruned filter. This method operates without the need for original training data, utilizing an oblivious selection algorithm that ensures the process remains simple, fast, and computationally efficient. By prioritizing this single-pass approach, the framework achieves a high level of sustainability while maintaining predictive performance comparable to major data-driven pruning benchmarks. The second theme, detailed in Chapter 4, investigates the structural impact of filter replacement within convolutional layers. While traditional pruning is often conceptualized as replacing a filter with a "zero-filter" (total elimination), this research considers the advantages of non-zero filter replacement. We identify specific conditions where substituting redundant filters with optimized, lower-dimensional alternatives leads to a significant reduction in model computations. This chapter introduces an optimization framework designed to find these non-zero substitutes, ensuring that the reduction in parameters does not compromise the network's representational fidelity. Finally, Chapter 5 provides a comparative study using the ThiNet framework as a baseline to evaluate the impact of different algorithmic selection paradigms. By investigating the performance of both greedy and oblivious algorithms, we demonstrate that the oblivious approach maintains parity in accuracy while being significantly faster and more resource-efficient. These findings serve as a fundamental motivation for the broad adoption of oblivious algorithms throughout this thesis, positioning them as a superior engineering choice for building fast, simple, and effective model optimization pipelines. In the following sections, we assess the technical strengths and limitations of these proposed methods as reflected in each of these research themes.

6.1 Data-Free Pruning via Analytical Error Bounds

This section introduces a novel pruning methodology rooted in the derivation of a formal upper bound on the Average Absolute Error propagated from a pruned layer to the subsequent feature maps. By moving beyond isolated layer analysis, this approach characterizes the "cascading impact" of filter removal on the network's internal manifold.

The methodology is entirely data-free, relying solely on the intrinsic algebraic properties of the weight tensors to identify redundancy. A key innovation within this framework is the implementation of an oblivious selection algorithm. Unlike greedy methods that require computationally expensive iterative re-inference, this single-pass ranking mechanism ensures the pruning process remains fast, efficient, and sustainable. Central to this algorithmic reliability is the formal proof that the proposed importance function satisfies the γ-weak property. This property provides a rigorous mathematical foundation to ensure that the oblivious selection process is guaranteed to converge. While the resulting architecture is sub-optimal compared to an exhaustive global search, the gap is theoretically bounded, ensuring that the performance remains highly competitive with state-of-the-art results.

Furthermore, the efficacy of the proposed method is validated through Resource Efficiency (RE), a newly defined metric that quantifies the total computational expenditure required to reach a specific accuracy threshold. Comparative analysis reveals that our approach is significantly more efficient regarding the total FLOPs consumed during the optimization lifecycle. While the proposed method recovers the accuracy of the pruned network to levels comparable with leading SOTA techniques, it does so using only a small fraction of the computational resources. This study has been accepted for publication in the IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI).

6.2 The Paradigm of Filter Replacement

This section shifts the research focus from traditional pruning-as-elimination toward a more nuanced strategy of filter replacement. While pruning is conventionally framed as substituting

a filter with a zero-filter (total removal), this work investigates the potential of non-zero replacement, which involves substituting complex filters with optimized, lower-dimensional alternatives.

In Chapter 4, we introduce a formal upper bound on the Average Absolute Error (AAE) of the succeeding layer, specifically for the context of filter substitution. The importance function developed for this selection process is directly derived from this theoretical bound. This chapter identifies specific architectural conditions where total filter elimination would result in a catastrophic loss of representational fidelity. In these scenarios, the proposed optimization framework derives a substitute that preserves the critical signal characteristics of the original filter at a significant reduction in computational cost. By mathematically demonstrating that this replacement paradigm minimizes FLOPs while maintaining higher structural integrity than total removal, this work provides a new pathway for aggressive model reduction without the typical accuracy trade-offs.

Furthermore, the results demonstrate that non-zero filter replacement achieves superior performance compared to established pruning methods, particularly in scenarios where fine-tuning is omitted from the compression procedure. The methodologies and findings detailed in this study have been peer-reviewed and published in the IEEE Transactions on Neural Networks and Learning Systems (TNNLS).

6.3 Comparative Analysis of Selection Complexity

This Theme studied in chapter 5 provides the empirical justification for the algorithmic choices made throughout the thesis, utilizing the ThiNet pruning framework as a primary baseline for evaluation. The study conducts a rigorous comparative analysis between greedy and oblivious selection paradigms to determine their respective trade-offs in accuracy and computational overhead.

The findings reveal a notable complexity-performance gap. Despite its iterative complexity and exhaustive search logic, the greedy algorithm offers no significant accuracy advantage over

the oblivious approach across the tested architectures. Conversely, the oblivious algorithm is shown to be substantially faster and more resource-efficient, drastically reducing GPU memory utilization and execution time during the selection phase.

This case study serves as a fundamental motivation for the broad adoption of oblivious algorithms throughout this thesis, positioning them as the superior engineering choice for building high-speed, simple, and environmentally sustainable model optimization pipelines. By demonstrating that a single-pass selection mechanism can maintain representational fidelity, this work establishes a new standard for efficient "Green AI" compression. The methodologies and findings detailed in this study have been peer-reviewed and published in IEEE Signal Processing Letters.

CONCLUSION AND RECOMMENDATIONS

7.1 General Conclusion

This thesis has addressed the critical intersection of deep learning optimization and environmental sustainability, specifically within the context of structural pruning for Convolutional Neural Networks. As the industry trends toward increasingly massive architectures, the "Green AI" paradigm has transitioned from a theoretical preference to a technical necessity. The central challenge addressed by this research was the reliance of modern pruning methods on vast amounts of data and immense computational power during the optimization phase. By shifting the perspective from empirical, weight-based heuristics to a rigorous analysis of functional inter-layer dependencies, this work has successfully demonstrated that high-performance model compression can be achieved in a data-free and resource-efficient manner.

The foundational pillar of this research was the mathematical modeling of the interdependency of the convolutional layers. By deriving analytical upper bounds for the Average Absolute Error propagated across adjacent layers, we established a principled justification for architectural reduction. This theoretical breakthrough allowed for the identification of filter redundancy based on the intrinsic manifold of the network's weights rather than external data samples. This autonomy is vital for the future of edge computing, where privacy constraints and limited bandwidth often prohibit the use of original training distributions for post-deployment optimization.

A significant outcome of the experimental investigations was the resolution of the complexity-performance gap between selection paradigms. Through a rigorous comparison of greedy and oblivious selection strategies, it was determined that the iterative complexity of greedy search offers no substantial accuracy benefit over a well-informed, single-pass oblivious algorithm. By mathematically proving that the importance function satisfies the γ-weak property, this research

provided the stability and convergence guarantees necessary to adopt high-speed selection mechanisms. This shift effectively eliminates the prohibitive search tax that has historically plagued structural pruning, making the pruning process itself as efficient as the resulting sparse model.

Furthermore, this work expanded the horizons of structural modification through the introduction of non-zero filter replacement. By moving beyond the binary choice of keeping or deleting a filter, we introduced a methodology to substitute complex filters with optimized, lower-dimensional alternatives. This approach proved particularly effective in preserving representational fidelity in sensitive architectural regions where traditional pruning would lead to catastrophic information loss. This contribution, combined with the development of the Resource Efficiency metric, provides a comprehensive framework for Green AI.

As future work, this research can be expanded in the following directions:

1. **Architectural Generalization:** Extending the proposed pruning and replacement frameworks to diverse neural architectures beyond CNNs, such as Transformers, Graph Neural Networks (GNNs), and Mamba-based models.
2. **Hybrid Compression Strategies:** Integrating quantization-aware techniques with structural pruning to achieve higher compression ratios and further optimize the deployment of models on extremely resource-constrained hardware.
3. **Scale and Modality Expansion:** Adapting the data-free pruning paradigms to Large Language Models (LLMs) and Vision-Language Models (VLMs), specifically addressing the complex attention-based dependencies inherent in these large-scale systems.
4. **Holistic Multi-Objective Optimization:** Developing a global compression framework that moves beyond layer-wise interdependency to a comprehensive view of the entire network. This approach will treat predictive performance, computational complexity, and the total

carbon footprint as a joint optimization problem to address the core challenges of sustainable AI.

7.2 Contributions

The contributions of this thesis are:

- The first contribution of this work is the development of a formal mathematical framework for quantifying structural interdependencies within neural networks. By deriving analytical upper bounds for the Average Absolute Error propagated between adjacent layers, this research provides a principled alternative to empirical, magnitude-based pruning heuristics. This theoretical foundation allows for the characterization of a filter's importance based on its functional contribution to the succeeding layer's feature maps, ensuring that the structural reduction process is grounded in the algebraic properties of the network's internal manifold.
- Connected to this mathematical foundation is the proof of the γ-weak property for the proposed importance function. This contribution provides the necessary and sufficient conditions for algorithmic stability and guarantees the convergence of the selection process. By establishing this property, the research mathematically justifies the transition from computationally expensive greedy searches to high-speed oblivious paradigms. It ensures that the single-pass selection mechanism remains robust and reliable, yielding sparse architectures that are theoretically bounded and practically effective.
- A third major contribution is the engineering of the low-complexity, oblivious selection algorithm. This innovation addresses the search tax of model optimization by eliminating the complexity inherent in iterative greedy methods. By implementing a single-pass ranking strategy, this work significantly reduces the temporal and computational overhead of the pruning complexity. This contribution makes structural compression viable for real-time applications and deployment on resource-constrained hardware, providing a fast and simple pipeline that maintains parity with state-of-the-art predictive accuracy.

- The thesis further contributes a novel structural paradigm through the introduction of optimal non-zero filter replacement. Moving beyond the binary keep-or-delete approach, this strategy substitutes redundant filters with mathematically optimized, lower-dimensional alternatives. This replacement framework preserves critical signal characteristics in sensitive architectural regions where total filter elimination would lead to catastrophic representational loss. By demonstrating that this paradigm maintains higher fidelity than traditional removal, especially in scenarios without post-pruning fine-tuning, this work establishes a new standard for aggressive model reduction.
- Finally, this research contributes to the operationalization of Green AI through the derivation of the Resource Efficiency metric and hardware-agnostic carbon footprint modeling. This standardized evaluation protocol allows for the quantification of the environmental impacts of the pruning process. By validating this comprehensive framework across diverse architectures like VGG, ResNet, and MobileNet in a completely data-free environment, this thesis provides a universal benchmark for sustainable, data-free, and high-performance model optimization.

LIST OF REFERENCES

Alwani, M., Madhavan, V. & Wang, Y. (2021). DECORE: Deep Compression with Reinforcement Learning. 2022 IEEE. *CVF Conference on Computer Vision and Pattern Recognition (CVPR)(2021)*, pp. 12339–12349.

;and Hans Peter Graf, H. L. A. K. I. D. H. S. (2017). Pruning Filters for Efficient ConvNets. *In International Conference of Learning Representation (ICLR)*.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K. & Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4), 834–848.

Chen, L., Feldman, M. & Karbasi, A. (2018). Weakly submodular maximization beyond cardinality constraints: Does randomization help greedy? *International Conference on Machine Learning*, pp. 804–813.

Chen, Z., Xiang, J., Lu, Y., Xuan, Q., Wang, Z., Chen, G. & Yang, X. (2024). Rgp: Neural network pruning through regular graph with edges swapping. *IEEE Transactions on Neural Networks and Learning Systems*, 35(10), 14671–14683.

Cheng, Y., Wang, D., Zhou, P. & Zhang, T. (2017). A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*.

Dai, B., Zhu, C., Guo, B. & Wipf, D. (2018). Compressing neural networks using the variational information bottleneck. *International Conference on Machine Learning*, pp. 1135–1144.

Das, A. & Kempe, D. (2011). Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. *arXiv preprint arXiv:1102.3975*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255.

Deutsch, F. & Deutsch, F. (2001). *Best approximation in inner product spaces*. Springer.

Ding, X., Ding, G., Han, J. & Tang, S. (2018). Auto-balanced filter pruning for efficient convolutional neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Ding, X., Zhou, X., Guo, Y., Han, J., Liu, J. et al. (2019). Global sparse momentum sgd for pruning very deep neural networks. *Advances in Neural Information Processing Systems*, 32.

El Halabi, M., Srinivas, S. & Lacoste-Julien, S. (2022). Data-efficient structured pruning via submodular optimization. *Advances in Neural Information Processing Systems*, 35, 36613–36626.

Fan, F., Su, Y., Jing, P. & Lu, W. (2021). A dual rank-constrained filter pruning approach for convolutional neural networks. *IEEE Signal Processing Letters*, 28, 1734–1738.

Frankle, J. & Carbin, M. (2019). The lottery ticket hypothesis: Finding sparse, trainable neural networks. *ICLR*.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4), 193–202.

Ganjdanesh, A., Gao, S. & Huang, H. (2024). Jointly training and pruning cnns via learnable agent guidance and alignment. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16058–16069.

Girshick, R., Donahue, J., Darrell, T. & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, 27.

Guo, S., Zhang, L., Zheng, X., Wang, Y., Li, Y., Chao, F., Wu, C., Zhang, S. & Ji, R. (2023). Automatic network pruning via hilbert-schmidt independence criterion lasso under information bottleneck principle. *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 17458–17469.

Han, S., Pool, J., Tran, J. & Dally, W. (2015). Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28.

He, K., Zhang, X., Ren, S. & Sun, J. (2016a). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

He, K., Zhang, X., Ren, S. & Sun, J. (2016b). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

He, Y. & Xiao, L. (2024). Structured pruning for deep convolutional neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*.

He, Y., Liu, P., Wang, Z., Hu, Z. & Yang, Y. (2019a). Filter pruning via geometric median for deep convolutional neural networks acceleration. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4340–4349.

He, Y., Liu, P., Wang, Z., Hu, Z. & Yang, Y. (2019b). Filter pruning via geometric median for deep convolutional neural networks acceleration. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4340–4349.

He, Y., Zhang, X. & Sun, J. (2017). Channel pruning for accelerating very deep neural networks. *Proceedings of the IEEE international conference on computer vision*, pp. 1389–1397.

Hong, R., Hu, Z., Liu, L., Wang, M., Yan, S. & Tian, Q. (2015). Understanding blooming human groups in social networks. *IEEE Transactions on Multimedia*, 17(11), 1980–1988.

Huang, Z. & Wang, N. (2018). Data-driven sparse structure selection for deep neural networks. *Proceedings of the European conference on computer vision (ECCV)*, pp. 304–320.

Joo, D., Yi, E., Baek, S. & Kim, J. (2021). Linearly replaceable filters for deep network channel pruning. *Proceedings of the AAAI conference on artificial intelligence*, 35(9), 8021–8029.

Krizhevsky, A., Hinton, G. et al. (2009). Learning multiple layers of features from tiny images.

Lannelongue, L., Grealey, J. & Inouye, M. (2021). Green algorithms: quantifying the carbon footprint of computation. *Advanced science*, 8(12), 2100707.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4), 541–551.

Lee, N., Ajanthan, T. & Torr, P. H. (2018). Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*.

Lee, S. & Song, B. C. (2024). Fast filter pruning via coarse-to-fine neural architecture search and contrastive knowledge transfer. *IEEE Transactions on Neural Networks and Learning Systems*, 35(7), 9674–9685.

Lei, P., Liang, J., Zheng, T. & Wang, J. (2022). Compression of convolutional neural networks with divergent representation of filters. *IEEE Transactions on Neural Networks and Learning Systems*, 35(3), 4125–4137.

Li, Y., van Gemert, J. C., Hoefler, T., Moons, B., Eleftheriou, E. & Verhoef, B.-E. (2023). Differentiable transportation pruning. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16957–16967.

Liebenwein, L., Baykal, C., Lang, H., Feldman, D. & Rus, D. (2020). Provable filter pruning for efficient neural networks. *ICLR*.

Lin, M., Ji, R., Wang, Y., Zhang, Y., Zhang, B., Tian, Y. & Shao, L. (2020). Hrank: Filter pruning using high-rank feature map. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1529–1538.

Lin, M., Cao, L., Li, S., Ye, Q., Tian, Y., Liu, J., Tian, Q. & Ji, R. (2021). Filter sketch for network pruning. *IEEE Transactions on Neural Networks and Learning Systems*, 33(12), 7091–7100.

Lin, S., Ji, R., Li, Y., Wu, Y., Huang, F. & Zhang, B. (2018). Accelerating Convolutional Networks via Global & Dynamic Filter Pruning. *IJCAI*, pp. 8.

Lin, S., Ji, R., Yan, C., Zhang, B., Cao, L., Ye, Q., Huang, F. & Doermann, D. (2019). Towards optimal structured cnn pruning via generative adversarial learning. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2790–2799.

Liu, G., Zhang, K. & Lv, M. (2022). Soks: Automatic searching of the optimal kernel shapes for stripe-wise network pruning. *IEEE Transactions on Neural Networks and Learning Systems*.

Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S. & Zhang, C. (2017). Learning efficient convolutional networks through network slimming. *Proceedings of the IEEE international conference on computer vision*, pp. 2736–2744.

Long, J., Shelhamer, E. & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440.

Luo, J.-H. & Wu, J. (2020). Neural network pruning with residual-connections and limited-data. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1458–1467.

Luo, J.-H., Zhang, H., Zhou, H.-Y., Xie, C.-W., Wu, J. & Lin, W. (2018). ThiNet: Pruning CNN filters for a thinner net. *IEEE transactions on pattern analysis and machine intelligence*, 41(10), 2525–2538.

Mariet, Z. & Sra, S. (2016). Diversity networks: Neural network compression using determinantal point processes. *ICLR*.

Naftali, T., C, P. F. & William, B. (2000). The information bottleneck method. *arXiv preprint physics/0004057*.

Nonnenmacher, M., Pfeil, T., Steinwart, I. & Reeb, D. (2022). SOSP: Efficiently capturing global correlations by second-order structured pruning. *ICLR*.

Park, M., Kim, D., Park, C., Park, Y., Gong, G. E., Ro, W. W. & Kim, S. (2024). Reprune: Channel pruning via kernel representative selection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(13), 14545–14553.

Peng, H., Wu, J., Chen, S. & Huang, J. (2019). Collaborative channel pruning for deep networks. *International conference on machine learning*, pp. 5113–5122.

Qian, Y., He, Z., Wang, Y., Wang, B., Ling, X., Gu, Z., Wang, H., Zeng, S. & Swaileh, W. (2023). Hierarchical threshold pruning based on uniform response criterion. *IEEE Transactions on Neural Networks and Learning Systems*.

Ren, S., He, K., Girshick, R. & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C. & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3), 211-252. doi: 10.1007/s11263-015-0816-y.

Sarvani, C., Ghorai, M., Dubey, S. R. & Basha, S. S. (2022). Hrel: Filter pruning based on high relevance between activation maps and class labels. *Neural Networks*, 147, 186–197.

Srinivas, S. & Babu, R. V. (2015). Data-free parameter pruning for deep neural networks. *arXiv preprint arXiv:1507.06149*.

Sui, Y., Yin, M., Xie, Y., Phan, H., Aliari Zonouz, S. & Yuan, B. (2021). Chip: Channel independence-based pruning for compact neural networks. *Advances in Neural Information Processing Systems*, 34, 24604–24616.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. & Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9.

Tang, X., Ye, S., Shi, Y., Hu, T., Peng, Q. & You, X. (2025). Filter Pruning Based on Information Capacity and Independence. *IEEE Transactions on Neural Networks and Learning Systems*.

Tian, G., Chen, J., Zeng, X. & Liu, Y. (2021). Pruning by training: A novel deep neural network compression framework for image processing. *IEEE Signal Processing Letters*, 28, 344–348.

Tishby, N. & Zaslavsky, N. (2015). Deep learning and the information bottleneck principle. *2015 ieee information theory workshop (itw)*, pp. 1–5.

Tofigh, S., Ahmad, M. O. & Swamy, M. (2022). A low-complexity modified thinet algorithm for pruning convolutional neural networks. *IEEE Signal Processing Letters*, 29, 1012–1016.

Wang, C., Grosse, R., Fidler, S. & Zhang, G. (2019). Eigendamage: Structured pruning in the kronecker-factored eigenbasis. *International conference on machine learning*, pp. 6566–6575.

Wang, Y., Lu, Y. & Blankevoort, T. (2020). Differentiable joint pruning and quantization for hardware efficiency. *European Conference on Computer Vision*, pp. 259–277.

Wang, Z., Liu, X., Huang, L., Chen, Y., Zhang, Y., Lin, Z. & Wang, R. (2022). Qsfm: Model pruning based on quantified similarity between feature maps for ai on edge. *IEEE Internet of Things Journal*, 9(23), 24506–24515.

Ye, J., Lu, X., Lin, Z. & Wang, J. Z. (2018). Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. *arXiv preprint arXiv:1802.00124*.

You, Z., Yan, K., Ye, J., Ma, M. & Wang, P. (2019). Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks. *Advances in neural information processing systems*, 32.

Yu, R., Li, A., Chen, C.-F., Lai, J.-H., Morariu, V. I., Han, X., Gao, M., Lin, C.-Y. & Davis, L. S. (2018). Nisp: Pruning networks using neuron importance score propagation. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9194–9203.

Yvinec, E., Dapogny, A., Cord, M. & Bailly, K. (2021). Red: Looking for redundancies for data-freestructured compression of deep neural networks. *Advances in Neural Information Processing Systems*, 34, 20863–20873.

Yvinec, E., Dapogny, A., Cord, M. & Bailly, K. (2023). Red++: Data-free pruning of deep neural networks via input splitting and output merging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3), 3664–3676.

Zheng, X., Ma, Y., Xi, T., Zhang, G., Ding, E., Li, Y., Chen, J., Tian, Y. & Ji, R. (2021). An information theory-inspired strategy for automatic network pruning. *arXiv preprint arXiv:2108.08532*.

Zhou, Y., Wei, T., Zhu, X. & Collin, M. (2021). A parcel-based deep-learning classification to map local climate zones from Sentinel-2 images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14, 4194–4204.

Zhuang, T., Zhang, Z., Huang, Y., Zeng, X., Shuang, K. & Li, X. (2020). Neuron-level structured pruning using polarization regularizer. *Advances in neural information processing systems*, 33, 9865–9877.

Zniyed, Y., Nguyen, T. P. et al. (2025). Enhanced network compression through tensor decompositions and pruning. *IEEE Transactions on Neural Networks and Learning Systems*, 36(3), 4358–4370.