T.C.
BAHCESEHIR UNIVERSITY
GRADUATE SCHOOL
THE DEPARTMENT OF ARTIFICIAL INTELLIGENCE
MASTER'S PROGRAM IN ARTIFICIAL INTELLIGENCE

OPTIMIZING CNNS VIA SØRENSEN-DICE BASED PRUNING

MASTER'S THESIS

MUSAB HAJBI

ISTANBUL 2025

T.C.
BAHCESEHIR UNIVERSITY
GRADUATE SCHOOL
DEPARTMENT OF ARTIFICIAL INTELLIGENCE
MASTER'S PROGRAM IN ARTIFICIAL INTELLIGENCE

OPTIMIZING CNNS VIA SØRENSEN-DICE BASED PRUNING

MASTER'S THESIS

MUSAB HAJBI

THESIS ADVISOR

PROF. DR. SÜREYYA AKYÜZ

ISTANBUL 2025

T.C.
BAHCESEHIR UNIVERSITY
GRADUATE SCHOOL

27/05/2025

MASTER THESIS APPROVAL FORM

Program Name:	Artificial Intelligence (English, Thesis)
Student's Name and Surname:	Musab Hajbi
Name Of The Thesis:	Optimizing CNNs via Sørensen-Dice based pruning
Thesis Defense Date:	27th May 2025

This thesis has been approved by the Graduate School which has fulfilled the necessary conditions as Master thesis.

Assoc. Prof. Yücel Batu SALMAN

Institute Director

This thesis was read by us, quality and content as a Master's thesis has been seen and accepted as sufficient.

	Title/Name	**Institution**	**Signature**
Thesis Advisor's	Prof. Dr. Süreyya AKYÜZ	Bahçeşehir University	
Member's	Assist. Prof.Tarkan AYDIN	Bahçeşehir University	
Member's	Assoc. Prof. Burcu TUNGA	İstanbul Technical University	

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: HAJBI MUSAB

Signature:

ABSTRACT

OPTIMIZING CNNS VIA SØRENSEN-DICE BASED PRUNING

Musab, Hajbi
Master's Program in Artificial Intelligence
Supervisor: Prof. Dr. Süreyya Akyüz

May 2025, 35 pages

Deep learning, particularly Convolutional Neural Networks CNNs, has revolutionized many fields, including image recognition, natural language processing, and object detection. However, the high computational complexity and memory requirements of these models hinder their deployment on resource-constrained devices. In this paper, a novel kernel pruning framework is proposed that leverages the robust Sørensen-Dice similarity coefficient to identify and remove redundant kernels within each convolutional layer. This method efficiently reduces model complexity while preserving essential structural information and accuracy. Extensive experiments on popular CNN architectures and benchmark datasets demonstrate the advantages of this approach over other pruning methods.

Key Words: Pruning CNNs, Sørensen-Dice Similarity, Model Optimization.

ÖZET

SØRENSEN-DICE TABANLI BUDAMA İLE CNNS'NİN OPTİMİZE EDİLMESİ

Musab, Hajbi
Yapay Zeka Yüksek Lisans Programı
Tez Danışmanı: Prof. Dr. Süreyya Akyüz

Mayıs 2025, 35 sayfa

Derin öğrenme, özellikle Evrişimli Sinir Ağları (CNN'ler), görüntü tanıma, doğal dil işleme ve nesne tespiti gibi birçok alanda devrim yaratmıştır. Ancak, bu modellerin yüksek hesaplama karmaşıklığı ve bellek gereksinimleri, kaynakları sınırlı cihazlarda kullanılmalarını zorlaştırmaktadır. Bu çalışmada, her bir evrişim katmanı içindeki gereksiz çekirdekleri belirlemek ve kaldırmak için güçlü Sørensen-Dice benzerlik katsayısını kullanan özgün bir çekirdek budama çerçevesi önerilmektedir. Bu yöntem, modelin yapısal bütünlüğünü ve doğruluğunu koruyarak model karmaşıklığını etkili bir şekilde azaltmaktadır. Popüler CNN mimarileri ve karşılaştırmalı veri kümeleri üzerinde yapılan kapsamlı deneyler, bu yaklaşımın diğer budama yöntemlerine kıyasla üstünlüklerini ortaya koymaktadır.

Anahtar Kelimeler: CNN Budama, Sørensen-Dice Benzerliği, Model Optimizasyonu.

This thesis is dedicated to all people who encouraged and supported through doing my thesis starting from my parents, professors and friends.

ACKNOWLEDGEMENTS

I wish to express my deepest gratitude to my supervisor Prof. Dr. SÜREYYA AKYÜZ for her guidance, advice, criticism, encouragements and insight throughout the research.

I would also like to thank my parents, Hasan HAJBI and Lara KASBI, for their great support throughout my life. Without their understanding, and continuous support, I could have never been able to aspire for this level of education and complete this study.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

Chapter 1

Introduction

Convolutional Neural Networks (CNNs) have revolutionized numerous fields, serving as the backbone for remarkable breakthroughs in machine learning. Their powerful feature extraction capabilities have driven advancements across a wide spectrum of applications, from enabling sophisticated autonomous navigation in vehicles to improving diagnostic accuracy in healthcare. The success of CNNs lies in their ability to automatically learn hierarchical representations directly from data, leading to unprecedented performance on tasks such as image recognition, object detection, and semantic segmentation. Most notably these hierarchical feature representations opened the path for transfer-learning, in which they can be used to "transfer knowledge" to different tasks and domains where data scarcity is a problem such as medical domains acting as a feature extraction stage.

While CNNs have clearly excelled, they are also known to be extremely overparametrized. While this helps with good generalization, it often incurs more complexity and a lot of functional redundancy, especially in the intermediate convolution potentially with many filters contributing to similar or repeating features which only results in increased size and computation with neither proportionate gain in predictive performance nor accuracy and likely has negative consequences for efficiency, interpretability of model deploy ability.

1.1 Statement of the Problem

Despite their remarkable accuracy and widespread adoption, the increasing complexity and scale of modern convolutional neural networks present significant practical challenges. High-capacity models containing millions or even billions of parameters, require a substantial amount computational resource as well as their large memory footprints and intensive processing power. This results in considerable hurdles for real-world deployment, particularly in environments with limited resources such as embedded systems, mobile devices, and applications requiring real-time inference. The consequences include prolonged training times, high inference latency, increased energy consumption, and substantial storage requirements. Addressing these limitations through efficient model optimization techniques is therefore crucial for enabling the broader deployment and accessibility of powerful CNN models.

A variety of model compression techniques, most notably pruning, have been proposed to address these issues. Pruning aims to remove any redundant or unimportant parameters from the trained network architecture reducing size and speeding up the inference as well.

Two major paradigms for paradigms appear as follow:

- ***Unstructured pruning*** aims to zero out individual weights, can achieve high compression ration but produces irregular sparsity that standard hardware and libraries cannot exploit efficiently.
- ***Structured pruning*** removes entire kernels/filters or channels, which in turn yield dense networks that map nicely into existing hardware. However most

structured approached rely on simple heuristic or require extensive manual tuning and they often overlook overlap between layers functionally.

An additional challenge with structured pruning is the fact that there is no consensus on what a true “redundant” filter is. Many of the popular structured pruning approaches use indirect proxies, such as weight magnitudes, or sparsity of feature maps. These heuristics are simple to compute but leaves plenty of room for error and can even remove filters that contain unique discriminative features that are subtle. In fact, many of these heuristics require trial and error tuning with each architecture and dataset, which also limits its use across datasets and new architectures. We need a more principled, similarity-aware approach, to detect a joint function amongst the filters, and to prune those that are truly redundant.

Given these limitations of existing pruning approaches, there is a clear need for more principled, automated techniques that can effectively identify and remove redundancy without relying on a lot of manual intervention or simple heuristics. This work confronts a central challenge in CNN optimization; “How to identify and remove both uninformative and redundant filters in CNNs in an automated way that avoids manual per-layer tuning”. By addressing this question, we aim to deliver compact, accurate, and hardware-friendly CNNs that can be deployed more broadly and selected more intelligently.

1.2 Theoretical Overview

Inspired by the application of similarity metrics in other domains for model optimization, particularly the successful use of the Sørensen-Dice index for pruning random forest ensembles by (Bonasera & Carrizosa, 2024) this study adapts the Sørensen-Dice similarity coefficient as a novel metric for pruning forest ensemble stability. The Sørensen-Dice coefficient, originally developed in ecology to measure the overlap between two sets (Dice, 1945) has proven robust in various applications, Its theoretical foundation in set similarity provides a distinct perspective compared to traditional vector-space or statistical correlation measures. By viewing the activation patterns or normalized weights of convolutional kernels as 'sets', the Sørensen-Dice index can theoretically capture the degree of overlap in the features they respond to, offering a principled way to identify functionally redundant kernels.

This similarity-based approach aligns well with the nature of overparameterized in CNNs, where many filters may exhibit near-identical activation patterns or having filter with unrepresentative features. Unlike magnitude-based or gradient-based criteria, the Sørensen-Dice index offers a symmetric and bounded measure of pairwise kernel similarity, ranging from 0 (no overlap) to 1 (perfect match). When used across all pairwise combinations within a layer, it provides a global picture of feature overlap, enabling the identification of filters that are semantically redundant. This allows for more informed pruning decisions that preserve representational diversity.

Furthermore, structured pruning introduces dependencies between layers; removing filters in one layer changes the required input shape for the subsequent convolutional layer. Maintaining a valid network architecture and ensuring that pruning decisions propagate correctly requires robust dependency management.

Theoretical frameworks, such as DepGraph (Fang, Ma, Song, Bi Mi, & Wang, 2023) provide mechanisms to track and manage these inter-layer structural relationships automatically, ensuring that pruning operations result in a coherent and executable network.

Synthesizing these theoretical components, this work proposes a structured pruning strategy grounded in the Sørensen-Dice similarity coefficient to quantify kernel redundancy or lack of representative features, complemented by dependency management to handle structural changes. This approach aims to move beyond heuristic-based pruning by providing a similarity-aware criterion for identifying redundant filters, thereby enabling more effective and automated CNN compression.

1.3 Purpose of the Study

The primary objective of this study is to develop and evaluate a kernel pruning framework that leverages the Sørensen-Dice similarity coefficient. By quantitatively assessing the similarity between convolutional kernels, the proposed method aims to selectively prune redundant and highly similar filters, thereby producing a more compact and efficient network. Furthermore, we extend the use of the Sørensen-Dice similarity coefficient as a model evaluation metric: by comparing multiple base models through their internal activation similarity profiles, we can identify which model retains richer, more diverse feature representations. This allows the selection of the most promising base model to serve as the starting point for transfer learning and fine-tuning, ensuring better generalization and adaptation across tasks.

1.4 Significance of the Study

This work makes several key contributions to the field of CNN optimization, particularly in model compression and pruning domains. By proposing a novel method to identify and address redundancy, this work aims to improve the efficiency and applicability of deep learning models especially in resource constrained domains. The specific contributions of this study can be summarized as follows:

1. **Novel Sørensen-Dice Similarity for Kernel Pruning**: Introduce the use of the Sørensen-Dice coefficient as a novel metric for quantifying convolutional kernel redundancy, offering a distinct perspective compared to traditional similarity measures.
2. **Automated Quantile-Based Thresholding**: Propose a novel, automated thresholding strategy using quantile functions to dynamically determine layer-specific pruning ratios, eliminating manual tuning.
3. **Redundancy-Aware Pruning via Representative Kernel Selection**: Leverage the proposed similarity metric and quantile-based thresholding to retain only representative kernels, discarding duplicated or overlapping feature extractors, enabling efficient pruning with fine-tuning performance comparable to that of full-model optimization.

Chapter 2

Literature Review

Convolutional neural networks (CNNs) have achieved remarkable accuracy on many vision tasks, but their depth and width often come with prohibitive storage and compute costs. For example, (Simonyan & Zisserman, 2015) showed that increasing depth to 16–19 layers (VGG nets) substantially improves ImageNet accuracy. However, such deep networks contain millions of parameters, which restricts deployments on resource-limited or edge devices. Consequently, a rich literature on *CNN pruning* has emerged, aiming to remove redundant weights or channels while preserving performance (Han, Mao, & Dally, 2016; He & Xiao, 2024). Pruning methods are broadly classified into unstructured (weight-wise) and structured (filter/channel-wise) approaches.

Early work by (Han et al., 2016) introduced "Deep Compression," a pipeline of pruning, quantization, and Huffman coding, has achieved up to 35x reductions in storage for models like AlexNet and 49x for VGG without accuracy loss . This weight-level pruning removes individual small-magnitude weights and then re-trains the network. Such magnitude-based pruning assumes that weights with smaller absolute values contribute least to accuracy. In practice, unstructured pruning yields irregular sparsity that hardware often cannot exploit, because it does not reduce the convolutional compute cost (Li, Kadav, Durdanovic, Samet, & Graf, 2017).

The increases interest in network pruning has motivated several surveys aimed at organizing the growing literature. (Cheng, Zhang, & Shi, 2024) present a detailed comprehensive taxonomy of deep neural network pruning structure in four main scopes. These scopes are **universal/specific speedup**, that includes unstructured, structured and semi-structured pruning methods, **when to prune,** which covers static pruning before, during and after training in addition to runtime pruning with settings like one-shot or iterative pruning approaches, **how to prune**, this involves pruning be certain criteria or learning, **fusion of pruning and other compression techniques,** and example of which is knowledge distillation in large language models (LLMs), vision models and multimodal models. In contrast domain focuses surveys emphasize the unique aspect of new architectures for example, (Tang et al., 2024) specifically focus on method for transformer models, highlighting their unique architecture which necessitates specific compression method that must handle the complex components present, such as multi-head attention and n feed-forward modules. Instead of just removing isolated weight removal in an unstructured manner, pruning transformers involves removing entire substructures to achieve hardware speed-up. Furthermore, pruning can also be combined by compression techniques such as quantization. Quantization simply refers to the reduction in number of bits used for weights or activations e.g. reducing representation from 32-bits to 16,8-bits only. However (Tang et al., 2024) notes that finding the proper combination strategy is complex and has high computational cost for transformers.

Structured pruning methods address this by removing entire filters or channels. (Li et al., 2017) propose pruning convolutional filters whose weights have small norm, which directly reduces feature-map dimensions. (Liu et al., 2017) impose channel-wise sparsity via scale factors (in batch normalization) during training and then prune channels with near-zero scales, yielding thin networks. (Li et al., 2017) also explored channel pruning schemes for modern architectures. These structured approaches produce pruned models that remain dense and hardware-friendly: removing whole filters does not create arbitrary sparsity patterns, so existing dense-matrix libraries can realize speedups. SqueezeNet (Iandola et al., 2016) suggests an alternative by designing a small network, it achieves AlexNet level accuracy with 50x fewer parameters , showing that careful architecture design can drastically reduce size.
Despite these advances, magnitude-based pruning has notable limitations. It treats each weight (or filter) independently and ranks them by simple criteria (e.g. absolute value), ignoring complex interdependencies. As pointed out in recent surveys, the assumption that smaller weights have least influence can be flawed in nonlinear networks (Cheng et al., 2024). Magnitude-based pruning often requires multiple pruning–retraining iterations to recover performance, since aggressively removing small weights can disrupt feature representations. Moreover, its reliance on fixed thresholds or hyperparameters means it is not globally optimized: one must hand-tune the pruning percentage for each layer or network. Bullet-list critiques include:

Neglect of inter-layer dependencies: Magnitude based methods do not account for how weights in different layers jointly affect outputs (Fang et al., 2023). Removing a weight in one layer may have non-obvious effects on later layers.
Irregular sparsity: Pruning individual weights yields sparse connections that do not translate to real speedup without special hardware. Structured alternatives show the benefit of removing complete filters to reduce operations (Li et al., 2017).
***Heuristic thresholds*:** Traditional methods need user-defined pruning ratios or threshold values per layer, which may not match the network's actual importance distribution. These hyperparameters are often chosen by trial and error.

One class of methods uses gradient-based sensitivity. For example, *SNIP* (Lee, Ajanthan, & Torr, 2019) and *GraSP* (C. Wang, Zhang, & Grosse, 2020) assess connection salience at initialization via gradients rather than weight magnitude. In particular, *GraSP* (Gradient Signal Preservation) scores weights by how pruning would preserve gradient flow through the network, pruning kernels whose removal least disrupts gradient propagation. These approaches implicitly account for interdependencies among weights by using loss derivatives to gauge importance. SNIP identified salient connection by computing the gradient of the loss using a binary mask over the weights, measuring each connections attribute to the loss in data-dependent way, enabling single-shot pruning without needing any pretraining or retraining cycles. *GraSP* on the other hand leverages second order information of the Hessian-gradient product in Taylor expansion to estimate impact of removing weights. By leveraging loss derivatives at initialization, both methods offer more functionally meaningful pruning criteria than traditional magnitude-based method, while avoiding the computational cost of training-based methods. Both approaches have their limitations, *SNIPs* reliance on a connection sensitivity criterion based on isolated gradient magnitude for each weight is suboptimal as it can shift after pruning, potentially leading to bottlenecks certain layers in the architecture at high sparsity while both

approaches may still underperform compared to pruning strategies applied to fully trained models.

Some studies have explored a different more principled approach to overcome the limitation of magnitude-base pruning, including similarity measures, regularization techniques and structured sparsity methods. One approach is Soft filter pruning (SFP) (He, Kang, Dong, Fu, & Yang, 2018) which uses the $l_{p\text{-norm}}$ to get soft-pruning candidates representing low importance filters, zeroes them out during iterative pruning while allowing them to updates. Representing a sparsity similar approach before finally removing any low filter after model converges to a structured result. However, this still requires training the entire iterative model before final kernel removal, introducing some computational overhead. Other approaches leverage a more sophisticated criteria for pruning such as geometric relationships . (Yao, Li, Kang, & Wang, 2022) proposed a channel pruning approach based on angular dissimilarity between filter DACP, combining group-lasso penalty with angular similarity constrains to achieve enhanced sparsity without the need for extra training. This however relies on basis vector approximation and authors acknowledge the need for more expressive formulations to guide sparsity. Similarly, (Fang, Ma, Mi, & Wang, 2024) introduced a Isomorphic pruning that groups sub-structure within a network by their topology and performs importance/ranking base on these groups. This method can handle various model architecture with novel mechanisms. Yet while this method yields competitive results across architectures, its effectiveness may be limited by factors beyond structural similarity, such as training strategy or regularization, which require further exploration.

The Dependency-Aware Pruning via DepGraph method proposed by (Fang et al., 2023) introduces a generalizable framework for structured pruning across diverse neural architectures, including CNNs, RNNs, GNNs, and Transformers Byway of constructing of a dependency-graph that is able to captures both inter-layer and intra-layer structural relationships, such as those induced by residual connections or shared pruning constraints between batch normalization and convolutional layers. By recursively propagating these dependencies, it identifies groups of structurally coupled parameters. This “coupling, which not only forces different layers to be pruned simultaneously but also expects all removed parameters to be consistently unimportant, thereby avoiding structural issues and significant performance degradation after pruning”.

Despite its versatility, DepGraph faces several limitations. Group-level importance estimation can be imprecise, especially when relying on simple norm-based metrics that may not reflect true parameter significance. Moreover, the sparse training phase introduces additional computational cost and complexity. In large-scale or highly entangled architectures, the resulting dependency graph can also grow considerably, making the grouping process and subsequent pruning more challenging. Additionally, as with many structured pruning approaches, retraining is often necessary to mitigate performance degradation post-pruning.

In contrast, the L_1-norm-based filter pruning method introduced by (Li et al., 2017) targets convolutional neural networks by eliminating entire filters with low L_1-norms, under the assumption that these filters contribute minimally to the learned

representations of the network. It is done by pruning filters and their corresponding feature maps, this approach introduces structured sparsity in the network architecture, which aligns well with existing acceleration libraries and hardware. Hence improving practical inference efficiency. One of the notable advantages is its simplicity, the pruning decision relies solely on the pre-trained weights, making it computationally inexpensive and data-independent during the pruning stage. However, this method also has some limitations. The assumption that smaller norm implies lower importance, but even filters with small norms can still encode valuable features. Furthermore, layers in CNNs exhibit varying levels of redundancy and sensitivity to pruning, requiring careful tuning of per layer pruning ratios. This method's applicability is largely restricted to convolutional architectures and does not inherently account for structural dependencies, limiting its effectiveness in networks with skip connections or shared parameters. As with (Fang et al., 2023), fine-tuning is typically required to restore model accuracy after pruning.

Some Recent work has addressed the limitations of heuristic pruning methods by shifting towards optimization-driven frameworks. (Ali, Üçüncü, Ataş, & Özöğür-Akyüz, 2020) introduce a novel ensemble pruning algorithm for the classification of Motor Imagery (MI) EEG signals, employing a Difference of Convex Algorithm (DCA) to optimize an objective function that balances both the accuracy and diversity of an ensemble of Support Vector Machines (SVMs). This approach, which had not been used on EEG data before, aims to select an optimal subset of classifiers, demonstrating improved performance with a reduced number of classifiers. Similarly, (Buse Çisil Güldoğuş, Abdullah, Ali, & Özöğür-Akyüz, 2023) propose a mathematical model based on sparse second-order cone programming (SOCP) to simultaneously optimize accuracy and diversity when pruning ensembles of Convolutional Neural Networks (CNNs). Their model directly obtains the pruning percentage and addresses the neglect of diversity in accuracy-focused pruning. Complementing these application-driven approaches, (Le Thi, Le, Phan, & Tran, 2021)provide a more theoretical exploration of DC programming and DCA for a special class of non-convex problems, developing standard and accelerated DCA schemes (ADCA) and introducing DCA-Like, which offers a new way to approximate the DC objective function without a strict DC decomposition. Their work includes a rigorous study of the convergence properties of these algorithms and an application to the t-distributed Stochastic Neighbor Embedding (t-SNE) problem in machine learning, even showing that the Majorization Minimization (MM) algorithm for t-SNE is a special case of DCA-Like. These diverse applications of mathematically rigorous optimization techniques, like DCA and SOCP, and works by other authors like (B.Ç. Güldoğuş & Özögür-Akyüz, 2025; Buse Çisil Güldoğuş et al., 2023; Otar & Akyuz, 2017; Özöğür-Akyüz, Otar, & Atas, 2020; Üçüncü, Akyüz, & Gül, 2024) ,underscore a clear trend towards principled methods for network and ensemble pruning.

Furthermore (Bonasera & Carrizosa, 2024) approach to extract interpretable rules from tree ensembles leverages the Sørensen-Dice index, to assess the stability of individual rule lists within the ensemble. It defines the stability of a rule list as the sum of the Sørensen-Dice indices computed between that list and all other lists in the random forest, effectively quantifying the number of shared splits. This similarity-driven approach forms a crucial part of their method, where the selection of rule lists is biased towards those with higher stability, indicating they are composed of rules that

are consistently replicated throughout the forest. (Bonasera & Carrizosa, 2024) argue that by maximizing this stability, the selected lists become less sensitive to data perturbation and are more likely to resemble a decision tree structure, thereby enhancing interpretability. Their method aims to provide a condensed, interpretable model that retains most of the predictive power of the full tree ensemble. Through rigorous computational experiments, they provide statistically significant evidence that their method is competitive with other rule extraction methods, suggesting that their similarity-driven selection can preserve accuracy effectively.

Alternative pruning strategies have emerged in response to the limitations of traditional magnitude-based or Hessian-informed techniques, with some aiming to refine the pruning schedule itself and others challenging the very notion of complexity in pruning criteria. One such method is growing regularization, proposed by (H. Wang, Qin, Zhang, & Fu, 2021), which reimagines regularization not as a fixed constraint, but as a dynamic process that intensifies throughout training. Unlike conventional sparse training methods that apply weak penalties uniformly or rely heavily on importance scores computed in advance, growing regularization begins with low penalty strength and incrementally amplifies it, thereby guiding unimportant weights toward zero in a progressive manner. This dynamic escalation of penalty implicitly captures second-order information, specifically local curvature without explicitly calculating the Hessian, making it computationally attractive while still leveraging deeper structural cues within the loss landscape.

The technique is represented in two algorithmic variants. First is the GReg-1, which emphasizes the timing of pruning rather than immediately removing weights identified as low-importance by L_1-norms, it applies a growing L_2 penalty, allowing these weights to decay gradually before being pruned in a controlled manner. This staged process improves the network's ability to maintain performance, especially when large portions of the model are removed. The second, GReg-2, improves the selection of weights for pruning by relying on the natural separation of weight magnitudes under increasing regularization. It uses a simple criterion like L_1-norm to effectively identify less expressive weights without the need for approximation methods or complex importance estimations, while also assigning negative penalties to retained weights to restore their expressive capacity post-pruning. Both methods support structured and unstructured pruning, and scale well to larger datasets and architectures.

Despite its advantages, growing regularization also present some challenges. The methods require predefined pruning ratios per layer, which could limit adaptability across architectures or tasks. Additionally, the use of multiple new hyperparameters, such as penalty ceilings and scheduling granularity, introduces more tuning complexity. The theoretical underpinnings of GReg-2 also rely heavily on L_2-specific assumptions, potentially limiting their generalizability to scenarios where L_1 or other forms of regularization might be preferred.

In contrast to this nuanced optimization of pruning mechanics, other work has taken a more provocative stance by questioning the very need for sophisticated pruning criteria. (Frankle & Carbin, 2019) through their exploration of the Lottery Ticket Hypothesis, introduce a framework wherein sparse subnetworks, termed "winning

tickets" can match the performance of the original dense networks if trained from their original initialization. These subnetworks are discovered through iterative magnitude pruning and retain their original weight values during fine-tuning. A surprising insight from their experiments is that when these pruned subnetworks are reinitialized randomly, their ability to reach comparable performance deteriorates significantly, particularly as sparsity increases. This highlights the critical role of initialization, suggesting that sparse structures alone are insufficient, how and when they were initialized matters equally.

Although their work does not propose random pruning as an endorsed method, their experiments model it through randomly sampled subnetworks and random reinitialization, showing that these generally underperform relative to structured pruning strategies. The apparent contradiction with findings from (Liu, Sun, Zhou, Huang, & Darrell, 2019), who observed that pruned and randomly reinitialized networks could still be trained successfully, is addressed by positing a threshold of sparsity. Within highly overparameterized models, random reinitialization may be tolerable at moderate sparsity levels, but only subnetworks with good initialization, those resembling winning tickets, remain competitive at high sparsity.

This implies a key limitation in the use of random pruning as a practical strategy: while randomness can occasionally produce trainable subnetworks, it does not systematically identify or preserve the crucial conditions (particularly initialization) that make a sparse subnetwork functionally viable. Thus, despite its apparent simplicity, random pruning is unreliable in practice and highly sensitive to initial conditions, especially when pushed toward higher degrees of sparsity.

The Sørensen-Dice similarity coefficient, first proposed in ecology to quantify species distribution overlap (Dice, 1945), is repurposed here as a dual-purpose metric for convolutional neural network (CNN) compression and transfer learning. Originally designed to measure habitat similarity where a score of 1 indicates identical species compositions and 0 signifies no overlap. This statistic's robustness to class imbalance and intuitive interpretability make it uniquely suited for deep learning applications. For instance, in medical imaging, the Sørensen-Dice index has become a gold standard for evaluating segmentation accuracy, ensuring precise alignment between predicted and ground-truth tumor boundaries (Ronneberger, Fischer, & Brox, 2015).

Building on this versatility, we propose adapting the Sørensen-Dice index to identifying redundant convolutional kernels for structured pruning. In the pruning setting, we view each filter's activation pattern across a representative dataset as a "distribution" whose overlap with others can be measured. Filters whose Sørensen-Dice index score exceeds a chosen threshold are deemed redundant; one representative from each redundant group is preserved, while the rest are removed. This process is inherently data-driven and parameter-light, relying only on the distributional overlap metric rather than handcrafted importance scores or resource-intensive optimization.

Chapter 3

Methodology

This chapter outlines the novel framework for pruning convolutional kernels based on the Sørensen-Dice similarity coefficient, along with the integration of a dependency graph framework DepGraph (Fang et al., 2023) to manage inter-layer dependencies during the pruning process.

3.1 Kernel Similarity Calculation

The primary goal of our method is to assess the similarity between convolutional kernels within each layer and remove those that are either uninformative or to similar and redundant. This on normalized kernel weights moving the values to a range of [0-1], this ensures the similarity calculation give a correct similarity score between the same range representing no similarity at zero up to identical at one.
To achieve this Each kernel's weights matrix is first normalized using absolute max normalization.

The choice of absolute max normalization as the normalization method is motivated because of its ability to preserve the relative contrasts and intensity patterns within each kernel, rather than artificially stretching or compressing the data as other normalization methods like Min-Max normalization might do. By scaling each kernel's weights according to their own maximum absolute value, it ensures that differences in amplitude and contrast-which are often meaningful for distinguishing kernel functionality-are retained in the normalized representation. This approach avoids the pitfalls of techniques like Min-Max normalization, which can mask these differences by forcing all kernels into the same range regardless of their original distribution, and it is less sensitive to outliers. As a result, absolute max normalization provides a more faithful basis for similarity calculations, leading to more reliable identification of redundant or uninformative kernels during pruning.

The normalization is computed as:

$$\widehat{w_i} = \frac{w_i}{\max(|w_i|)}, \tag{1}$$

where $(\widehat{w_i})$ is the normalized kernel weights, and $(\max(|w_i|))$ is the absolute maximum value within each of the respective kernels. This normalization step serves to preserve the relative structure and spatial relationships within each kernel making the similarity computation scale-invariant and reducing the impact of outlier weights on the similarity assessment and ensuring consistent comparisons across different layers of the network. This normalization scales kernel weight to the desired [-1 to 1].

However, since the Sørensen-Dice similarity operates on non-negative inputs, the normalized weights are subsequently rescaled to the [0, 1] interval using:

$$\widehat{w_i} \leftarrow \frac{\widehat{w_i} + 1}{2} \tag{2}$$

This rescaling ensures compatibility with the similarity formula and allows for consistent element-wise comparisons.

Once the kernel weights are normalized, similarity computation between two kernels **A** and **B**, each of size $k \times k$, is performed using the Sørensen-Dice similarity index. This index is used to measure the similarity between two sets and is computed as:

$$S = \frac{2 \cdot |\mathrm{A} \cap \mathrm{B}|}{|\mathrm{A}| + |\mathrm{B}|}, \tag{3}$$

where $\boldsymbol{S}$ is the similarity score between the two sets $\boldsymbol{A}$ and $\boldsymbol{B}$, and $|A \cap B|$ represents the intersection of the two sets, while $|A|$ and $|B|$ denote the magnitudes of sets A and $\boldsymbol{B}$, respectively.

In the context of convolutional kernel pruning, each kernel is treated as a set of normalized weights. The similarity between two kernels **A** and **B**, each of size $k \times k$, is computed by applying the Sørensen-Dice formula to their flattened and normalized weight values. Specifically, the similarity score between kernel i and kernel j is calculated as:

$$S_{ij} = \frac{2.\sum min\left(\widehat{w_i}, \widehat{w_j}\right)}{\sum \widehat{w_i} + \sum \widehat{w_j}}, \tag{4}$$

where $\widehat{w_i}$ and $\widehat{w_j}$ represent the normalized weights of kernels $\boldsymbol{A}$ and $\boldsymbol{B}$, respectively. The numerator $2.\sum min\left(\widehat{w_i}, \widehat{w_j}\right)$ represents the element-wise intersection between the two kernels, calculated by taking the minimum value of the corresponding elements from each kernel and summing them. The denominator $\sum \widehat{w_i} + \sum \widehat{w_j}$ the total sum of all elements in kernels $\boldsymbol{A}$ and $\boldsymbol{B}$**,** respectively, representing their magnitudes.

This formula provides a measure of similarity between the two kernels based on the proportion of overlapping elements (where the value in each kernel is similar) compared to their overall sum. Higher similarity scores indicate that the kernels are more similar, and lower scores indicate less similarity.

3.2 Proposed Pruning Approach

Based on the computed similarity scores, our pruning process follows two key rules low average similarity and high average similarity bounds. Both rules are calculated using Quantile-based automated thresholding. First in Low average similarity, kernels with an average similarity below a dynamically computed threshold τ_{min} (determined using a quantile function to remove the lowest x% of

similarity scores) are identified. These kernels are considered uninformative and are marked for pruning.

The other type of kernels to be removed are high similarity kernels with an average similarity above a dynamically computed threshold τ_{max} (determined using a quantile function to remove the highest x% of similarity scores). These kernels are considered redundant or learning duplicate/identical features and are also marked for pruning.

Alongside candidate kernel selection a "limiter "is introduced by ways of a fixed pruning ratio p limiter is enforced per layer, where no more than $p * N$ number kernels are pruned. This ensures that sufficient model capacity is preserved. Both thresholds τ_{min} and τ_{max} are computed after the absolute max normalization step to ensure that the retained kernels are both informative and distinct.

By applying the Sørensen-Dice similarity index in a layer-wise pruning framework, our method can robustly identify and remove redundant kernels, reducing model complexity while maintaining—or even improving—accuracy. Specifically, we compute all pairwise Dice similarities within each convolutional layer, then follow a greedy selection strategy akin to (Li et al., 2017) dynamically accounting for filters already discarded in preceding steps. At each iteration, the filter whose average overlap with all other kernels is highest (i.e., most redundant) or lowest (i.e., least informative) relative to the dynamically computed thresholds is pruned. After removal, we immediately recompute Similarity so that subsequent Dice scores reflect the updated kernels set. Because we continue this process until a user-defined performance drop target is reached, no fixed pruning ratios are required; the algorithm naturally adapts its intensity based on observed redundancy.

This layer-wise, similarity-driven pruning yields several tangible benefits:

1. Statistical Rigor: Dice directly measures informational overlap, avoiding hand-tuned importance heuristics.
2. Dependency-Aware Greediness: Early-layer pruning decisions propagate gracefully to deeper layers, preserving critical hierarchical features.
3. Hardware Friendliness: Entire convolutional kernels are removed (not zeroed), enabling efficient mapping onto accelerators without sparse matrix overhead.
4. Accuracy Preservation: Even under aggressive compression, our Sørensen-Dice pruned networks match or exceed standard baselines by focusing on uniquely informative filters. After some fine-tuning

The pruning Algorithm Implementation process is summarized in Algorithm 1

Algorithm 1 Sørensen-Dice-Based Kernel Pruning

Input: Trained CNN model $\mathcal{M}$, pruning ratio $p \in [0,1]$
Output: Pruned CNN model $\mathcal{M}'$
for each convolutional layer $\mathcal{L} \in \mathcal{M}$ **do**
  $Let\ \mathcal{W} = \{w_i\}_{i=1}^{N}$ be the set of N kernels in $\mathcal{L}$
  for each kernel $w_i \in \mathcal{W}$ **do**
    Normalize: $\widehat{w_i} = \frac{w_i}{\max(|w_i|)}$
    Rescale: $\widehat{w_i} \leftarrow \frac{\widehat{w_i}+1}{2}$
  end for
  Compute similarity matrix $\boldsymbol{S} \in R^{\mathrm{N}\times\mathrm{N}}$ where:

$$S_{ij} = \frac{2.\sum min(\widehat{w_i}, \widehat{w_j})}{\sum \widehat{w_i} + \sum \widehat{w_j}},$$

  Compute average similarity score:

$$\frac{1}{N-1}\sum_{i\neq j} S_{ij}$$

  Estimate thresholds using quantiles:

$$\tau_{min} = Q_\alpha(\{S_{\mathrm{avg},i}\}_{i=1}^{N}), \quad \tau_{max} = Q_\beta(\{S_{\mathrm{avg},i}\}_{i=1}^{N})$$

  Identify low-similarity kernels: $\mathcal{K}_{low} = \{i \mid S_{avg,i} < \tau_{min}\}$
  Identify high-similarity kernels: $\mathcal{K}_{high} = \{i \mid S_{avg,i} > \tau_{max}\}$
  Merge candidates: $\mathcal{K}_{low} \cup \mathcal{K}_{high}$
  Prune kernels in $\mathcal{K}$ from layer $\mathcal{L}$, update temporary model $\mathcal{M}'$
end for
Return final pruned model $\mathcal{M}'$

Following candidate kernel selection kernel, it is crucial to choose an appropriate and kernel removal approach. Since the goal is to reduce inference time and computational costs, zeroing candidate kernel is excluded, and a full removal of the candidate kernels and their dependencies is needed.

Due to the inherent inter-layer dependencies in convolutional neural networks, pruning kernels in one layer can alter the structure and kernel shapes in subsequent layers. To maintain structural consistency, our method integrates a dependency graph framework (Fang et al., 2023). This framework automatically manages the propagation of changes across layers, ensuring that the pruned model remains valid and that subsequent similarity computations correctly account for the altered network architecture.

Chapter 4

Experimental setup and Results

This chapter presents and analyzes the experimental results from two perspectives:

1- The evolution of accuracy as the pruning ratio increases.
2- Performance at a fixed pruning ratio after fine-tuning.

These highlight the trade-off between accuracy, inference speed, and memory efficiency, when applying the different pruning methods.

To evaluate the effectiveness of the proposed Sørensen–Dice-based pruning technique, experiments were conducted using popular CNN architectures: ResNet-50, VGG-19 and custom-built CNN depicted by

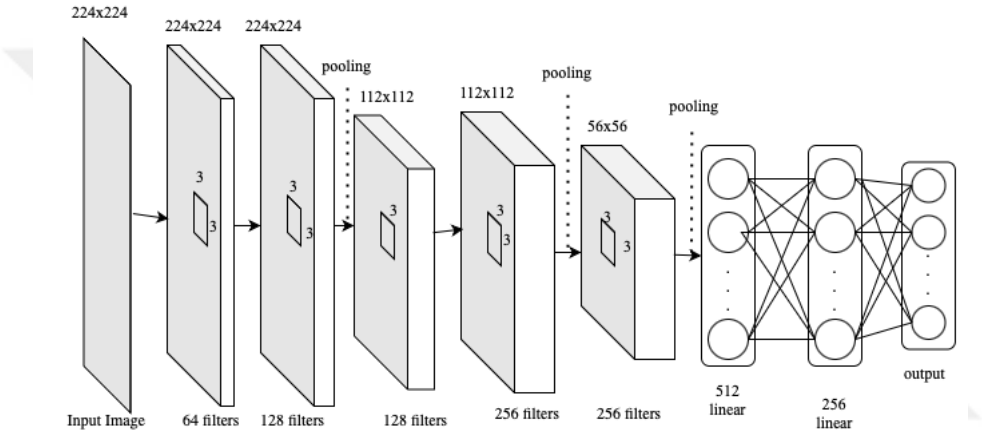


Figure 1. Custom CNN architecture.

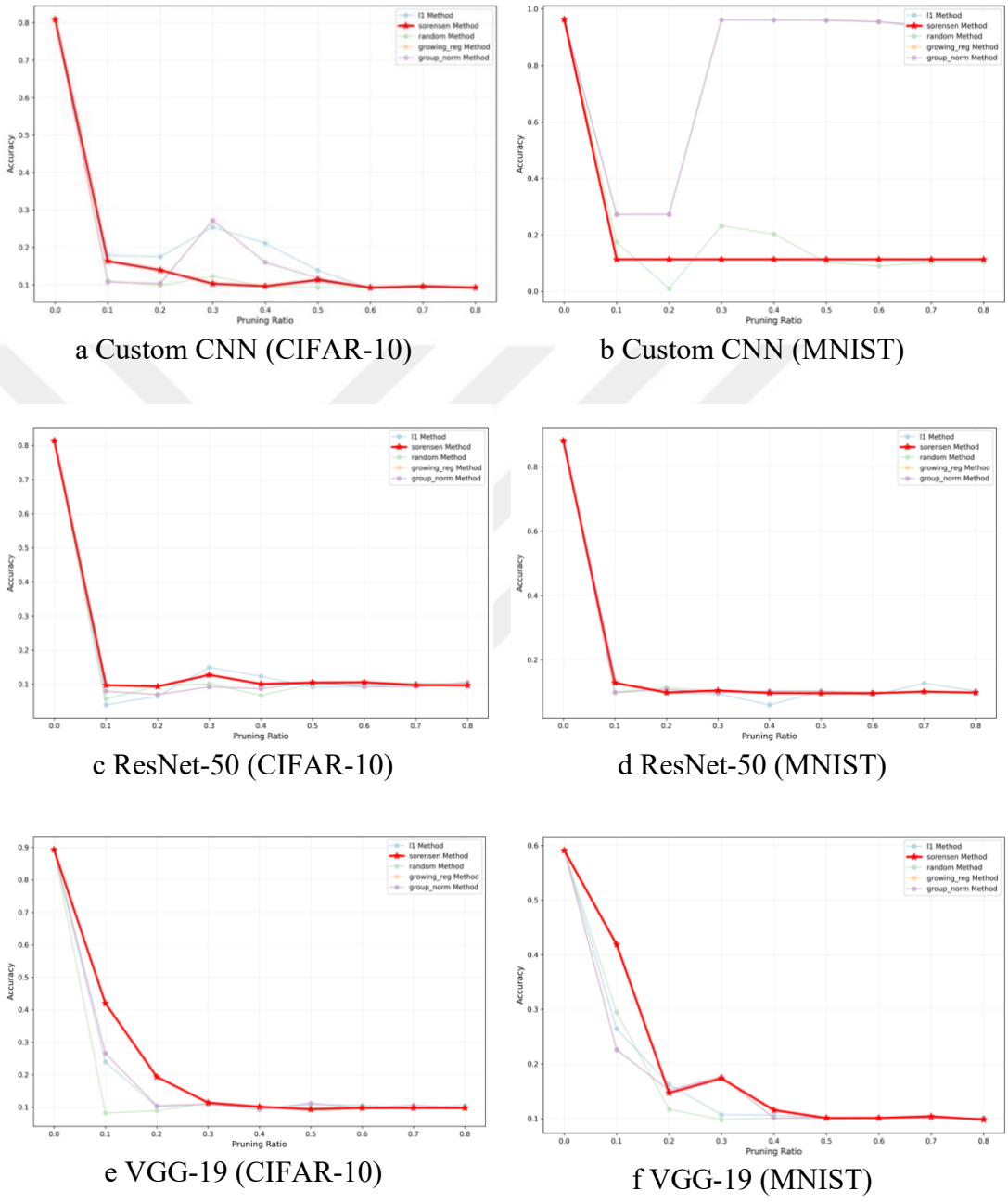
The Custom CNN in *Figure 1* was trained from scratch on each respective dataset while, the other two architectures were pretrained on ImagNet-1K dataset and adapted via transfer-learning to both CIFAR-10 and MNIST. Table 1 shows hyperparameters used for initial model fine-tuning as well pruned-models training.

Table 1

Hyperparameters

Parameter	Value
Batch Size	32
Optimizer	Adam
Learning Rate	1e-2 to 1e-4
Weight Decay	1e-5
Number of Epochs	20
Learning Rate Decay (gamma)	0.1
Loss Function	Cross-Entropy Loss
Data Augmentation	Horizontal flip, Random rotation

The experimental set up follows a consistent structure for both datasets, First the base models were obtained and, their accuracies were recorded (see *Figure 2* at zero pruning ratios). Next, a suitable pruning ratio value was selected, and each model was pruned using multiple pruning methods. Finally, the pruned models were fully trained on each respective dataset.



a Custom CNN (CIFAR-10) b Custom CNN (MNIST)

c ResNet-50 (CIFAR-10) d ResNet-50 (MNIST)

e VGG-19 (CIFAR-10) f VGG-19 (MNIST)

Figure 2. Accuracy vs pruning ratio trends.

Figure 2 illustrates how model accuracy changes with increasing pruning ratios across various methods (without fine-tuning). A significant initial decline in accuracy is observed as pruning becomes more aggressive.

This drop in accuracy, although noticeable, is justifiable. The rationale behind this trade-off is that pruning reduces redundancy and improves efficiency which is a key factor for deployment in resource-constrained environments. Further details on recoverability and the reasoning for pruning despite the initial accuracy drop are further discussed later. Additionally, while the accuracy drop might seem steep initially, it provides insight into the model's redundancy characteristics. Models with flatter accuracy drop curves across increasing pruning ratios exhibit higher internal redundancy and therefore are more amenable to compression. Conversely, sharp accuracy losses after mild pruning suggest limited redundancy and higher sensitivity to kernel removal, which was notably observed in VGG-19.

Based on these observations, we next compare accuracy at a fixed pruning ratio. In addition to evaluating accuracy, we also test computational efficiency; to do so we use the speed-up factor, calculated as:

$$Speed\text{-}up = \frac{Base\ FLOPS}{Pruned\ FLOPS}. \tag{5}$$

Here, FLOPs (Floating Point Operations) measure computational workload. Reducing FLOPs through pruning reduces inference time while aiming to maintain accuracy. A higher speed-up factor indicates greater computational efficiency.

Table 2 present results across different pruning methods at a fixed pruning ratio of (0.2), identified as the "elbow point" from *Figure 2*. This point, marks where accuracy degradation begins to accelerate. While pruning initially reduces accuracy, fine-tuning helps recover much of the lost performance, improving efficiency.

While our method was implemented as described in Algorithm 1 we also evaluated its performance by comparing it against several existing pruning approaches. Specifically, we consider $\boldsymbol{L_1}$ by (Li et al., 2017), **Growing Reg** by (H. Wang et al., 2021), **Random pruning** by (Frankle & Carbin, 2019) and **Group Norm** by (Fang et al., 2023) and their implementation of all methods on GitHub repo. A detailed comparison of the proposed Sørensen-Dice pruning method against several standard pruning techniques, across two datasets (CIFAR-10 and MNIST) and three network architectures (Custom CNN, ResNet-50, and VGG-19). Overall, the Sørensen-Dice method consistently achieves superior accuracy while maintaining competitive though sometimes slightly lower speed-up benefits compared to other methods.

Examining the Custom CNN results on CIFAR-10, the conventional pruning techniques (L_1, Random, Growing Reg, and Group Norm) yield accuracies clustered around 64–65% with a speed-up factor of 1.72. In sharp contrast, the Sørensen-Dice method achieves a markedly higher accuracy of 81%, with only a slight drop in speed-up to 1.55. A similar trend is observed for MNIST, where traditional methods again converge at 79% accuracy with a speed-up of 1.72, whereas Sørensen-Dice dramatically improves accuracy to 96%, with a marginally reduced speed-up of 1.50. These results highlight that the Sørensen-Dice approach is particularly well-suited for lightweight custom architectures, significantly boosting performance without heavily sacrificing computational gains.

In the case of ResNet-50 on CIFAR-10, the traditional methods achieve accuracies in the range of 64–67% with a speed-up of 1.64. The Sørensen-Dice method not only maintains a much higher accuracy of 79% but even improves the speed-up to 2.28, demonstrating its capability to enhance both accuracy and efficiency for deeper architectures. On MNIST, ResNet-50 sees a significant increase in accuracy around the 92% with traditional pruning to a huge 98% with Sørensen–Dice, with a moderate also a huge increase in speed-up from 1.64 to 2.28.

Table 2

Performance Comparison of Fixed Pruning Ratio

Dataset	**Model**	**Method**	**Test Accuracy%**	**F1-score**	**Speed-up**
CIFAR-10	Custom CNN	L_1 (Li et al., 2017)	64	0.62	1.72
		Random(Frankle & Carbin, 2019)	64	0.64	1.72
		Growing Reg(H. Wang et al., 2021)	65	0.64	1.72
		Group Norm(Fang et al., 2023)	64	0.65	1.72
		Sørensen-Dice(ours)	**81**	**0.79**	**1.55**
	ResNet-50	L_1(Li et al., 2017)	65	0.61	1.64
		Random(Frankle & Carbin, 2019)	64	0.57	1.64
		Growing Reg(H. Wang et al., 2021)	67	0.62	1.64
		Group Norm	65	0.63	1.64
		Sørensen-Dice(ours)	**79**	**0.79**	**2.28**
	Vgg-19	L_1(Li et al., 2017)	75	0.752	1.64
		Random(Frankle & Carbin, 2019)	74	0.745	1.64
		Growing Reg(H. Wang et al., 2021)	75	0.753	1.64
		Group Norm(Fang et al., 2023)	74	0.743	1.64
		Sørensen-Dice(ours)	**93**	**0.93**	**1.55**

Table 2 (cont.d)

MNIST	Custom CNN	L_1(Li et al., 2017)	79	0.79	1.72
		Random(Frankle & Carbin, 2019)	79	0.79	1.72
		Growing Reg	79	0.79	1.72
		Group Norm	79	0.79	1.72
		Sørensen-Dice(ours)	**96**	**0.96**	**1.55**
	ResNet-50	L_1(Li et al., 2017)	92.6	0.924	1.64
		Random(Frankle & Carbin, 2019)	92.8	0.927	1.64
		Growing Reg(H. Wang et al., 2021)	92.5	0.925	1.64
		Group Norm(Fang et al., 2023)	92.4	0.924	1.64
		Sørensen-Dice(ours)	**98**	**0.98**	**2.28**
	Vgg-19	L_1(Li et al., 2017)	94	0.945	1.64
		Random(Frankle & Carbin, 2019)	96	0.96	1.64
		Growing Reg(H. Wang et al., 2021)	94	0.943	1.64
		Group Norm(Fang et al., 2023)	94	0.943	1.64
		Sørensen-Dice(ours)	**98**	**0.97**	**1.55**

For VGG-19, the pattern varies slightly. On CIFAR-10, traditional methods achieve 74–75% accuracy with a speed-up of 1.64, whereas Sørensen-Dice dramatically lifts accuracy to 93%, with a small decrease in speed-up to 1.55. However, on MNIST, the Sørensen-Dice method achieves 98% accuracy, which is a substantial increase over other methods (80%). The speed-up with Sørensen-Dice for VGG-19 on MNIST is 1.55, slightly lower than the 1.64 observed for standard methods.

It is also worth noting that transfer learning played a key role in the performance observed on ResNet-50 and VGG-19. Pretrained weights from ImageNet provided a robust initialization that made these models more resilient to pruning-induced degradation, particularly when combined with our similarity-aware method. This supports our hypothesis that Sørensen-Dice-based pruning is well suited not only for models trained from scratch but also for fine-tuned pretrained models, as it preserves semantically rich features across domains.

The evaluation is extended by constraining every pruning method to operate exclusively on Conv2d layers. This restriction allows us to measure exact accuracy gains when speed up is exact between all methods and lets us measure exact throughput gains from filter removal alone. Because Sørensen-Dice selects kernels via quantile thresholds, we raised the pruning ratio slightly to (0.25) to allow exact speed ups.

We also stabilize the pruned networks, we reset Batch-Norm running statistics before fine-tuning, following the procedure recommended by (Humble, Shen, Latorre, Darve, & Alvarez, 2022) to freeze all weights, perform a few hundred forward passes, then begin. This one-minute recalibration step greatly improved convergence for every method, but the benefit is most visible for similarity-aware pruning, which otherwise distorts activation distributions more than magnitude-based baselines. Table 3 shows improvement in both accuracy and speed-up values.

Table 3

Convolution Layer Only Pruning

Dataset	**Model**	**Method**	**Test Accuracy%**	**F1-score**	**Speed-up**
CIFAR-10	Custom CNN	L_1 (Li et al., 2017)	68	0.68	1.76
		Random(Frankle & Carbin, 2019)	60	0.60	1.76
		Growing Reg(H. Wang et al., 2021)	70	0.71	1.76
		Group Norm(Fang et al., 2023)	70	0.71	1.76
		Sørensen-Dice(ours)	**75**	**0.75**	**1.76**
MNIST	Custom CNN	L_1(Li et al., 2017)	95	0.95	1.76
		Random(Frankle & Carbin, 2019)	56	0.57	1.76
		Growing Reg	95	0.95	1.76
		Group Norm	95	0.95	1.76
		Sørensen-Dice(ours)	**98**	**0.98**	**1.76**

We can observe that this approach allows us to recover and, in some cases, surpass the original model's accuracy. The pruned models in some cases recover around 80-90% of baseline accuracy and sometimes can match or surpass the unpruned baseline with 1.76x speed-up. This pruning/full parameter-tuning takes only around 60 minutes to calculate per method on an RTX 4090 GPU in contrast to 75 minutes on average for baseline model training.

Constraining all pruning techniques to Convolution-only operations equalizes the theoretical speed-up across methods and isolates the effect of each criterion on accuracy. Under this stricter setting Sørensen-Dice continues to outperform the magnitude- and randomness-based baselines, despite the slightly higher pruning ratio (0.25 versus the 0.20 "elbow" used in Table 2). On CIFAR-10 the similarity-aware

strategy lifts the Custom-CNN's test accuracy to 75 %, five percentage points above the best competing method and fifteen above random pruning, while maintaining the same throughput improvement. On MNIST, where most techniques already approach ceiling performance, Sørensen-Dice still adds three percentage points, showing that its filter-selection rule preserves salient features even in near-saturated regimes.

The gains are not simply a consequence of pruning fewer parameters. When compared with Table 2 where speed-up varied with pruning scope—Sørensen-Dice traded a small fraction of efficiency for large jumps in accuracy. In Table 3, with speed-up held constant, it remains ahead, demonstrating that the similarity metric identifies genuinely redundant filters rather than merely retaining more capacity. This finding supports the hypothesis that kernels exhibiting extreme Sørensen-Dice similarity (either highly redundant or uniquely dissimilar) are safe to drop, while mid-range kernels carry distinctive information essential for generalization.

Overall, Table 3 confirms that the Sørensen-Dice strategy scales gracefully from lightly pruned, mixed-layer scenarios to more aggressive values. By pairing similarity-aware pruning with a brief Batch-Norm reset, we achieve equal computational savings or even exceed baseline accuracy, validating the method's robustness across datasets, architectures, and pruning granularities.

In summary, the Sørensen-Dice pruning method proves especially effective on more complex and larger datasets (CIFAR-10), and on more compact or shallower architectures (Custom CNN), offering substantial accuracy gains with only modest trade-offs in speed-up. In some cases, such as ResNet-50 on CIFAR-10, it even surpasses conventional techniques in both accuracy and efficiency simultaneously, showcasing the robustness and adaptability of the proposed strategy across different architectures and tasks

Chapter 5
Conclusions

This chapter provides a comprehensive synthesis of the experimental findings and theoretical insights developed throughout this work, with particular focus on the effectiveness and implication of advance pruning strategies in deep learning. Building upon the results including those summarized in Table 2, the discussion explores how different pruning methods, especially the Sørensen-Dice approach, affect model performance across a variety of model architectures and datasets. Also, beyond evaluation traditional metrics such as computational efficiency, the chapter goes into the balance between preserving representational power and reducing complexity of the models, a trade-off that has quite the importance for deployment in resource constrained environments, transfer learning and model reusability. It also reflects on critical limitations observed in post-pruning retraining approach.

The results demonstrate that pruning strategies can achieve very different outcomes depending on how they prioritize computational reduction versus the preservation of important features. The Sørensen-Dice method shows a more selective pruning behavior: it reduces computational burden less aggressively than conventional methods but does so in a way that retains crucial kernels responsible for maintaining the model's predictive power. This balance is especially evident in the Custom CNN architecture, where the Sørensen-Dice method yields a significant boost in accuracy across both the CIFAR-10 and MNIST datasets. Even in the case of ResNet-50, a model with a more rigid, layered structure often seen in transfer learning applications, the method manages to deliver noticeable improvements, particularly on the MNIST dataset. These observations suggest that preserving key kernels can substantially mitigate the adverse effects typically associated with pruning, even within deeper and more complex networks.

An important aspect emerging from the experiments is the trade-off between computational speed-up and model accuracy. While traditional pruning techniques achieve higher speed-up factors (e.g., 1.72× and 1.64×), they often do so at the cost of substantial accuracy degradation. In contrast, the Sørensen-Dice method offers a more balanced trade-off, securing higher accuracy while only moderately compromising on speed-up. This balance is critical in real-world applications where even slight improvements in model performance can outweigh gains in computational efficiency. Scenarios such as autonomous driving, healthcare diagnostics, and security systems place a premium on precision, making methods that preserve model integrity highly valuable.

Beyond immediate performance metrics, the findings have important implications for broader areas such as neural architecture search and transfer learning. Neural architecture search, which seeks to automate the discovery of optimal network designs, can benefit significantly from pruning strategies that prioritize preserving the model's representational capacity rather than simply minimizing the number of parameters. Integrating informed pruning methods like Sørensen-Dice into NAS (Nueral Architecture Search) pipelines could lead to architectures that are both efficient and resilient, especially when deployed in resource-constrained environments.

Similarly, in transfer learning, where pre-trained networks are adapted to new tasks, aggressive pruning often leads to catastrophic performance drops. The results from ResNet-50 experiments reinforce the idea that careful, selective pruning helps maintain performance during transfer, making approaches like Sørensen-Dice particularly well-suited for fine-tuning models across different domains.

A further critical insight observed from our experiments is that pruning these different architectures then following up the pruning by training a linear classifier only, as don't in normal transfer learning, is insufficient to fully recover the performance protentional of some network architectures. While retraining a linear classifier may offer a quick and computationally inexpensive way to test viability of the retrained pruned models, it fails to reconfigure network weights in a manner that fully adapt to the new architecture, which is expected as the reconfiguration of network layer by layer affects the dependencies of weights in the network. A possible solution to be explored, is training multiple linear layers. As part of the classification head. Still this limitation becomes increasingly apparent as pruning rations grows, especially in shallower network like VGG and with more complex datasets. The experiment shows that a full re-training or at least a targeted fine-tuning of the entire network is essential to recalibrate the internal feature representations and to restore performance. In other word the pruning and retraining should be treated as a tightly coupled optimization process rather than a sequential one which we like to explore in further research.

Overall, these insights reinforce the idea that pruning should not solely be evaluated based on efficiency metrics like speed-up factors. Instead, future pruning strategies must carefully balance computational savings with the preservation of the model's ability to capture and represent complex patterns. By doing so, it becomes possible to develop deep learning models that are not only lighter but also smarter, more adaptable, and ultimately better suited for the growing range of real-world challenges they are expected to tackle

In summary, this thesis contributes to deep learning research in several important ways.

Robust Accuracy Preservation: The Sørensen-Dice method consistently ranks among the top approaches for maintaining high accuracy across datasets and models, particularly excelling with the Custom CNN,

Balanced Efficiency: Although its speed-up factor is marginally lower than that of other methods, the improved accuracy suggests that the computational cost is justified—especially in applications where precision is paramount,

Architecture Dependency: The varying results between Custom CNN and ResNet-50 emphasize the need for adaptive pruning. In transfer learning, aggressive pruning may harm performance, making methods like Sørensen–Dice, which selectively prune kernels, essential.

Overall, these insights demonstrate that while the conventional metrics of computational efficiency are important, preserving the integrity of the model's feature representations plays a critical role in maintaining accuracy. The Sørensen-Dice method offers a promising solution by effectively balancing these competing objectives.

Chapter 6
Discussion, Limitation And Future Outlook

This chapter builds on the findings discussed in the previous chapter, validating the code hypothesis of using a Sørensen-Dice-based pruning framework. It shows its efficacy in effectively balance accuracy preservation with computational efficiency gains performance, highlights methodological constraints, and outlines promising directions for future research and practical deployment.

6.1 Discussion

The results presented in this work confirm the efficacy of using the Sørensen-Dice-based pruning as a feature-aware kernel selections strategy that is able to maintain produce a substantial model pruning with highly recoverable accuracy across various datasets and model architectures, consistently outperforming standard magnitude-based pruning approaches, especially on architectures fully trained from scratch on source target dataset and on deeper model architectures like ResNet-50.

A particularly worthy observation is the superior results achieved on Resnet-50 under our pruning approach. This advantage my stem from the residual architecture itself and its high representation density and inherent redundancy, that makes it specially affected by similarity-based pruning. The Sørensen-Dice index can capture subtle overlaps between kernels, in Resnet, these overlaps are more pronounced due to the nature of its depth and present of skip connections, which allows the method to identify and remove redundant kernels more effectively. In contrast, with shallower networks with less overlap in kernel representation exhibit reduced pruning performance and are more sensitive to kernel removal, showing a tighter trade-ff between pruning and performance retention as observed with the VGG architecture.

Another point to note is the importance of the quantile-based thresholding mechanism, which eliminates the need for static, hand-tuned thresholds. Through using quantiles to determine upper and lower similarity bounds dynamically per layer, the method gains a level of autonomy and generalization across different architectures.

6.2 Limitations

Despite the advantages, the Sørensen-Dice-based approach carries several practical limitations that must be acknowledges. First, the method requires the manual specification of a global pruning ration for the network. Although an adaptive pruning threshold per-layer is implemented, the overall pruning target must be predefined, typical by selecting a ration that balances accuracy and compression. In our experiments, this was done using an elbow-point analysis on test data accuracy without fine-tuning. While effective, this approach involves evaluating multiple candidate ratios on hold-out data, which may not be feasible in scenarios where validation data is scarce or where automated pipeline integration is desired.

Second, the method assumes the need for a post-pruning phase, since the removal of entire convolutional kernels affect the learned representations, making it necessary to retrain and restore performance. Fine-tuning also involves scheduling additional training epochs and access to the full training dataset, which may not always be practical in some scenarios.

Third, while the layer-wise pruning mechanism is greedy and adaptive, the pruning workflow remains complex. The need to tune global pruning ratios, run elbow-point test, and perform fine-tuning across configurations, this introduces non-trivial overhead and opens the door to human error. A more integrated and automated pruning pipeline would reduce this burden and enhance reproducibility and usability in production workflows. Additionally, the current method operates exclusively on convolutional kernels. While this is appropriate for CNN-based architectures, its design and assumptions may not generalize to models where redundancy is more context-dependent or distributed, such as in attention-based mechanisms or recurrent architectures. Further abstraction and generalization of the pruning criterion are therefore needed.

Lastly, the Sørensen-Dice metric itself, though effective in identifying redundant kernels based on their similarity, does not explicitly account for downstream representational influence. That is, a kernel may appear redundant in isolation but contribute crucial non-linear interactions with subsequent layers. Future iterations should explore hybrid metrics that combine similarity with contribution analysis, such as SHAP or activation-based saliency scores.

With these encouraging results, several avenues for future research emerge. A natural next step would involve evaluating the Sørensen-Dice pruning strategy on larger and more diverse datasets, such as ImageNet or COCO, to assess its scalability and generalization ability. Expanding its application to a broader range of network architectures -including transformer-based models and graph neural networks- would further show the versatility and robustness of the approach across modern deep learning paradigms.

6.3 Future Work

Given the strong performance observe of the proposed method, several promising directions emerge for future research and development.

A logical next step is to evaluate the Sørensen-Dice pruning strategy on large-scale datasets such as ImageNet or COCO to test its scalability and generalization to real-world applications. These datasets pose challenges in terms of input complexity, class imbalance, and task diversity, which would provide a more rigorous benchmark for the method’s robustness.

Further research should also explore applying the technique to architectures beyond convolutional neural networks. Transformer models and graph neural networks are increasingly dominant across vision, language, and multimodal tasks. Investigating how Sørensen-Dice- based pruning performs in these architectures could validate its broader applicability across diverse deep learning paradigms.

Another promising direction lies in the integration of pruning with transfer learning. In particular, the Sørensen-Dice metric could be repurposed to measure feature alignment between source and target domains in pre-trained models. This would allow for more informed selection of base models in transfer learning pipelines by assessing kernel activation similarity rather than relying solely on brute force or trial-and-error. Such an approach could increase both the interpretability and efficiency of transfer learning in data-scarce scenarios.

In addition, combining the Sørensen-Dice framework with optimization technique, such as those used in (Bonasera & Carrizosa, 2024; Buse Çisil Güldoğuş et al., 2023; Miao et al., 2023) which may lead to hybrid strategies that jointly optimize pruning and accuracy. Removing the need for manual parameter selection and exploring this as an optimization problem. This could enable end-to-end pruning systems that are both theoretically grounded and empirically robust.

Finally, expanding the domain of application beyond classification task ins an interesting area of investigation. Tasks such as regression. object detection, segmentation, image captioning, and time-series forecasting would allow evaluation of the method's utility in multi-output and sequence-dependent settings. These tasks often require a more nuanced understanding of spatial and temporal features, offering a richer testbed for pruning strategies.

Ultimately, the promising performance of the Sørensen-Dice method as a novel pruning approach underscores its potential for practical, real-world deployment in environments where both speed and accuracy are critical. Whether in embedded systems, mobile applications, or large-scale industrial deployments, adopting feature-aware pruning strategies can enable the development of deep learning models that are not just lighter and faster, but also significantly more intelligent and trustworthy.

As deep learning continues to evolve toward greater complexity and broader applicability, the need for smarter model optimization techniques will only become more pressing. By demonstrating the value of informed, feature-preserving pruning, this thesis lays a foundation for future research aimed at creating models that truly balance performance and efficiency in a meaningful way.

REFERENCES

Ali, M. A., Üçüncü, D., Ataş, P. K., & Özöğür-Akyüz, S. (2020). Classification of Motor Imagery Task by Using Novel Ensemble Pruning Approach. *IEEE Transactions on Fuzzy Systems*, *28*(1), 85–91. https://doi.org/10.1109/TFUZZ.2019.2900859

Bonasera, L., & Carrizosa, E. (2024, June 30). *A Unified Approach to Extract Intepretable Rules from Tree Ensembles via Integer Programming*. arXiv. Retrieved from http://arxiv.org/abs/2407.00843

Cheng, H., Zhang, M., & Shi, J. Q. (2024, August 9). *A Survey on Deep Neural Network Pruning-Taxonomy, Comparison, Analysis, and Recommendations*. arXiv. https://doi.org/10.48550/arXiv.2308.06767

Dice, L. R. (1945). Measures of the Amount of Ecologic Association Between Species. *Ecology*, *26*(3), 297–302. https://doi.org/10.2307/1932409

Fang, G., Ma, X., Mi, M. B., & Wang, X. (2024, July 5). *Isomorphic Pruning for Vision Models*. arXiv. Retrieved from http://arxiv.org/abs/2407.04616

Fang, G., Ma, X., Song, M., Bi Mi, M., & Wang, X. (2023). DepGraph: Towards Any Structural Pruning. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 16091–16101. Vancouver, BC, Canada: IEEE. https://doi.org/10.1109/CVPR52729.2023.01544

Frankle, J., & Carbin, M. (2019, March 4). *The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks*. arXiv. https://doi.org/10.48550/arXiv.1803.03635

Güldoğuş, B.Ç., & Özögür-Akyüz, S. (2025). FSOCP: Feature selection via second-order cone programming. *Central European Journal of Operations Research*, *33*(1), 51–64. Scopus. https://doi.org/10.1007/s10100-023-00903-y

Güldoğuş, Buse Çisil, Abdullah, A. N., Ali, M. A., & Özöğür-Akyüz, S. (2023, February 12). *Autoselection of the Ensemble of Convolutional Neural Networks with Second-Order Cone Programming*. arXiv. https://doi.org/10.48550/arXiv.2302.05950

Han, S., Mao, H., & Dally, W. J. (2016, February 15). *Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding*. arXiv. https://doi.org/10.48550/arXiv.1510.00149

He, Y., Kang, G., Dong, X., Fu, Y., & Yang, Y. (2018, August 21). *Soft Filter Pruning for Accelerating Deep Convolutional Neural Networks*. arXiv. Retrieved from http://arxiv.org/abs/1808.06866

He, Y., & Xiao, L. (2024). Structured Pruning for Deep Convolutional Neural Networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *46*(5), 2900–2919. https://doi.org/10.1109/TPAMI.2023.3334614

Humble, R., Shen, M., Latorre, J. A., Darve, E., & Alvarez, J. (2022). Soft Masking for Cost-Constrained Channel Pruning. In S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, & T. Hassner (Eds.), *Computer Vision – ECCV 2022* (pp. 641–657). Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-20083-0_38

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016, November 4). *SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size*. arXiv. https://doi.org/10.48550/arXiv.1602.07360

Le Thi, H. A., Le, H. M., Phan, D. N., & Tran, B. (2021). Novel DCA based algorithms for a special class of nonconvex problems with application in

machine learning. *Applied Mathematics and Computation*, *409*, 125904. https://doi.org/10.1016/j.amc.2020.125904

Lee, N., Ajanthan, T., & Torr, P. H. S. (2019). *SNIP: SINGLE-SHOT NETWORK PRUNING BASED ON CONNECTION SENSITIVITY*.

Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H. P. (2017, March 10). *Pruning Filters for Efficient ConvNets*. arXiv. https://doi.org/10.48550/arXiv.1608.08710

Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., & Zhang, C. (2017, August 22). *Learning Efficient Convolutional Networks through Network Slimming*. arXiv. https://doi.org/10.48550/arXiv.1708.06519

Liu, Z., Sun, M., Zhou, T., Huang, G., & Darrell, T. (2019, March 5). *Rethinking the Value of Network Pruning*. arXiv. https://doi.org/10.48550/arXiv.1810.05270

Miao, M., Yang, Z., Zeng, H., Zhang, W., Xu, B., & Hu, W. (2023). Explainable cross-task adaptive transfer learning for motor imagery EEG classification. *Journal of Neural Engineering*. https://doi.org/10.1088/1741-2552/AD0C61

Otar, B. C., & Akyuz, S. (2017). *Ensemble clustering selection by optimization of accuracy-diversity trade off*. Presented at the 2017 25th Signal Processing and Communications Applications Conference, SIU 2017. Scopus. https://doi.org/10.1109/SIU.2017.7960636

Özöğür-Akyüz, S., Otar, B. Ç., & Atas, P. K. (2020). Ensemble cluster pruning via convex-concave programming. *Computational Intelligence*, *36*(1), 297–319. https://doi.org/10.1111/coin.12267

Ronneberger, O., Fischer, P., & Brox, T. (2015, May 18). *U-Net: Convolutional Networks for Biomedical Image Segmentation*. arXiv. https://doi.org/10.48550/arXiv.1505.04597

Simonyan, K., & Zisserman, A. (2015, April 10). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv. https://doi.org/10.48550/arXiv.1409.1556

Tang, Y., Wang, Y., Guo, J., Tu, Z., Han, K., Hu, H., & Tao, D. (2024, February 5). *A Survey on Transformer Compression* (Version 1). Version 1. arXiv. https://doi.org/10.48550/arXiv.2402.05964

Üçüncü, D., Akyüz, S., & Gül, E. (2024). A novel auto-pruned ensemble clustering via SOCP. *Central European Journal of Operations Research*, *32*(3), 819–841. Scopus. https://doi.org/10.1007/s10100-023-00887-9

Wang, C., Zhang, G., & Grosse, R. (2020). *PICKING WINNING TICKETS BEFORE TRAINING BY PRESERVING GRADIENT FLOW*.

Wang, H., Qin, C., Zhang, Y., & Fu, Y. (2021, April 5). *Neural Pruning via Growing Regularization*. arXiv. https://doi.org/10.48550/arXiv.2012.09243

Yao, J., Li, P., Kang, X., & Wang, Y. (2022, October 29). *A pruning method based on the dissimilarity of angle among channels and filters*. arXiv. https://doi.org/10.48550/arXiv.2210.16504