UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Innovative model pruning of Convolutional Neural Networks using the Hausdorff Distance

Diploma Thesis

Chrysoula Strifti

Supervisor: Dimitrios Katsaros

June 2024

UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Innovative model pruning of Convolutional Neural Networks using the Hausdorff Distance

Diploma Thesis

Chrysoula Strifti

Supervisor: Dimitrios Katsaros

June 2024

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Καινοτόμο Κλάδεμα Μοντέλου Συνελικτικών Νευρωνικών Δικτύων με τη χρήση της Hausdorff Απόστασης

Διπλωματική Εργασία

Χρυσούλα Στρίφτη

Επιβλέπων/πουσα: Δημήτριος Κατσαρός

Ιούνιος 2024

Approved by the Examination Committee:

Supervisor **Dimitrios Katsaros**
Associate Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Dimitrios Rafailidis**
Associate Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **George Thanos**
Laboratory Teaching Staff, Department of Electrical and Computer Engineering, University of Thessaly

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

«Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism».

The declarant

Chrysoula Strifti

Diploma Thesis

Innovative model pruning of Convolutional Neural Networks using the Hausdorff Distance

Chrysoula Strifti

Abstract

Over the past few years, deep neural networks have evolved and spread across various fields at an accelerated rate, claiming an increasingly significant position in our lives. However, deeper models, such as large language models (LLMs), require a tremendous amount of runtime and substantial computational capacity to be implemented. Only a few people have access to such powerful and expensive hardware, excluding a large portion of them from research and job opportunities in this specific domain and limiting them to a solely theoretical understanding of deeper architectures. This is due to the fact that a single model with billions of parameters requires multiple highly capable GPUs or even a supercomputer to be trained.

Pruning is the practice of reducing the size and complexity of neural networks by removing redundant parameters, enabling deep networks to run efficiently on low-resource machines, weaker GPUs, and even machines equipped only with CPUs for model training. Thus, pruning techniques for the optimization of neural networks have gained significant popularity. The goal of this work is to present a novel pruning technique, Hausdorff Distance based Pruning for Convolutional Neural Networks (HaDiPCS), that prunes filters inside convolutional layers of a model after calculating their dissimilarity based on the Hausdorff Distance between their weights. This technique achieves a considerable reduction in the floating point operations (FLOPs), with the maximum drop being 1/6 of the original model's FLOPs, while in most cases maintaining or even augmenting the accuracy of the model after pruning.

Keywords:

Filter Pruning, Hausdorff distance, Convolutional layers, CNNs, Deep Learning, Neural Networks, Filter dissimilarity

Διπλωματική Εργασία

Καινοτόμο Κλάδεμα Μοντέλου Συνελικτικών Νευρωνικών Δικτύων με τη χρήση της Hausdorff Απόστασης

Χρυσούλα Στρίφτη

Περίληψη

Τα τελευταία χρόνια, τα βαθιά νευρωνικά δίκτυα έχουν εξελιχθεί και εξαπλωθεί σε διάφορους τομείς με επιταχυνόμενο ρυθμό, διεκδικώντας ολοένα και πιο σημαντική θέση στη ζωή μας. Ωστόσο, τα βαθύτερα μοντέλα, όπως τα μεγάλα γλωσσικά μοντέλα (Large Language Models), απαιτούν εξαιρετικά μεγάλο χρόνο εκτέλεσης και σημαντική υπολογιστική ικανότητα για να υλοποιηθούν με τη χρήση πολλαπλών ικανότατων GPU ή ακόμη και υπερυπολογιστών. Μόνο λίγοι άνθρωποι έχουν πρόσβαση σε τόσο ισχυρό και ακριβό υλικό, αποκλείοντας ένα μεγάλο μέρος τους από την έρευνα και τις ευκαιρίες απασχόλησης στον συγκεκριμένο τομέα, περιορίζοντάς τους σε μια αποκλειστικά θεωρητική κατανόηση των βαθύτερων αρχιτεκτονικών. Το Pruning είναι η πρακτική της μείωσης του μεγέθους και της πολυπλοκότητας των νευρωνικών δικτύων με την αφαίρεση περιττών παραμέτρων, επιτρέποντας στα βαθιά δίκτυα (Deep Neural Networks) να εκτελούνται αποτελεσματικά σε μηχανήματα με χαμηλούς πόρους, ασθενέστερες GPU ή ακόμη και σε μηχανήματα εξοπλισμένα μόνο με CPU για την εκπαίδευση ενός μοντέλου. Έτσι, οι τεχνικές Pruning για τη βελτιστοποίηση των νευρωνικών δικτύων έχουν αποκτήσει σημαντική δημοτικότητα. Ο στόχος της παρούσας εργασίας είναι να παρουσιάσει μια νέα τεχνική pruning με όνομα, κλάδεμα Νευρωνικών Δικτύων με χρήση της Hausdorff απόστασης (HaDiPCS), που κάνει prune τα φίλτρα μέσα στα convolutional layers ενός μοντέλου αφού υπολογίσει τη διαφορετικότητά τους με βάση την απόσταση Hausdorff μέσω των βαρών τους. Η τεχνική αυτή επιτυγχάνει σημαντική μείωση των πράξεων κινητής υποδιαστολής (FLOPs), με τη μέγιστη πτώση να είναι το 1/6 των FLOPs του αρχικού μοντέλου, ενώ στις περισσότερες περιπτώσεις διατηρεί ή και αυξάνει την ακρίβεια του μοντέλου μετά το pruning και μειώνει σημαντικά το Inference time του μοντέλου.

Λέξεις-κλειδιά:

Pruning φίλτρων, Hausdorff απόσταση, συνελικτικά στρώματα, CNNs, Βαθιά Μάθηση, Νευρωνικά Δίκτυα, Ανομοιότητα Φίλτρων

Table of contents

List of figures

List of tables

Abbreviations

HaDiPCS	Hausdorff Distance based Pruning for Convolutional Neural Networks
HD	Hausdorff Distance
LLM	Large Language Model
etc	et cetera

Chapter 1

Introduction

Deep Learning has evolved tremendously in recent years extending its use to several applications across numerous fields. Autonomous cars, Large Language Models and Biomedicine applications are only a few of the domains Neural Networks expanded into. However, particularly vast computational resources are required for training and deploying larger and deeper models which has become an immense obstacle for an increasing amount of programmers and researchers that do not have access to such powerful hardware limiting their participation to cutting edge applications.

1.1 Thesis Subject

This thesis specifically targets the problem of reducing the computational complexity and resource requirements of deep neural networks through effective pruning techniques. This way deeper architectures can be implemented even in low-resource machines and even without a powerful GPU. This is achieved through, Hausdorff Distance based Pruning for Convolutional Neural Networks (HaDiPCS), a novel approach that systematically prunes filters within convolutional layers by measuring their dissimilarity based on the Hausdorff Distance between their weights. This method not only reduces the model's trainable parameters significantly, by up to 1/6 of the original model's parameters, which indicates a considerable decrease in the memory footprint of the model as well as the model's Floating Point Operations (FLOPs) but also aims to maintain or even improve the model's accuracy post-pruning.

1.1.1 Contributions

The contribution of the thesis is summarized as follows:

1. Several pruning techniques were researched and examined for their success
2. For the goal of the creation of a new pruning technique, diverse mathematical concepts were studied
3. A novel pruning technique was created based on the filter dissimilarity that results from the Hausdorff Distance calculation of the filter weights inside each layer
4. The pruning of an entire model was achieved using this novel technique
5. The HaDiPCS pruning was applied in a model using various datasets to establish its performance
6. Its high efficiency was concluded in both FLOPs and inference time reduction in the final pruned model when compared with the baseline model of each dataset
7. Possible Future optimizations of the algorithm were examined and presented

1.2 Structure of Thesis

A general introduction to Deep Learning and advancements in this field are going to be presented in Chapter 2 and more specifically the most popular Large Language Models and Neural Networks in Biomedicine are going to be evaluated. In Chapter 3 pruning is defined, the categories to which pruning is divided into are listed and several pruning techniques are briefly analyzed. In Chapter 4 the novel HaDiPCS technique is introduced, after the analysis of the Hausdorff distance for the understanding of the specific method. Moreover this Chapter describes all the steps followed for the pruning of a model based on the HaDiPCS technique. The results of several experiments are presented in Chapter 5 in order to assess the HaDiPCS technique as for its accuracy and robustness. Lastly, in Chapter 6 the main conclusions of the HaDiPCS technique are displayed as well as possible future directions for its optimization and further experimentation.

Chapter 2

Modern Developments in Deep Learning

2.1 Introduction to Deep Learning

While Deep learning has entered our lives in recent years it has a long history dating back to 1943 when Walter Pitts and Warren McCulloch inspired by the way typical neurons make connections during the thought process of the human brain, created the artificial neuron. Years later, Deep learning has evolved to unprecedented points due to the large disposal of data that are currently available for the training of Deep learning models. The robustness of data available facilitated the development of neural networks in various fields. Large Language Models (LLMs), automated vehicles and Neural network models in medicine are only a few of the domains which deep learning has revolutionized. This chapter will discuss the various advancements of Deep learning in two main fields, LLMs and neural networks in biomedicine outlining its advantages but also pointing out its weaknesses.

2.2 Large Language Models

2.2.1 Introduction to Large Language Models

Large Language Models (LLMs) are large-scale deep learning models, trained in a vast amount of data with an extreme volume of parameters, whose primary functionality is generating text that mimics human language. The various data used for the training process contain different styles of text and expressions such as slang, idioms and even localisms, thus the LLMs are able to produce versatile content and adapt to various linguistic tasks and conver-

sations while preserving a coherence in the text, without the need to be retrained.

The core neural network architecture of LLMs is Transformers, an architecture originally developed by Google's scientists published in the paper "Attention Is All You Need" in 2017 [1] which incorporated multi-head attention mechanisms building upon earlier work that introduced attention mechanisms [2] and suggested assigning varying weights on embeddings based on the attention humans show in different words inside a sentence.

The most significant applications of LLMs are text generation, translation, summarization, but nowadays their usage has extended to education and even healthcare and security. Famous examples of flexible LLMs are ChatGPT [3], Gemini [4] and pi ai [5], an interesting model that offers greater human-like interaction, with their functionalities spreading across all domains mentioned above.

2.2.2 Evaluation of Large Language Models

LLMs offer several possibilities, one of which is text generation and it is used for various purposes. LLMs can generate text of various styles and formats such as professional or creative writing and their capabilities in these writing tasks maintain a steadily increased performance and high quality.

While LLMs are proven to correspond well on data they are trained on and in general provide the user with correct information and great analysis of theories and concepts even in scientific fields such as physics, they will confidently answer incorrectly on several occasions but most commonly when provided with questions that required connection of various notions [6]. More specifically based on papers that test ChatGPT's competence on several academic fields, it is concluded that although it can present eloquently mathematical and physics concepts, it cannot create new knowledge nor answer correctly mathematical problems while it seems to understand the steps that need to be followed for a correct solution and even worse, it can provide different answers after each time it is asked the same question. In the field of chemistry LLMs can generally answer simple practical questions surpassing their respective ability in the field of physics. Research is not widely performed to assess LLMs' capabilities in these fields, thus results cannot entirely be available. In the field of math, it seems that LLMs can efficiently manage simple calculations such as addition, subtraction and multiplication while they also show competence in handling large, decimal and irrational numbers. But the more complicated the mathematical operation the more challenging its calculation

becomes for LLMs, with trigonometry and logarithmic functions being only some examples of operations LLMs perform incorrectly.

Another field where LLMs hold promising prospects is text classification with great response to problems such as sentiment analysis and genre identification while it also performs well with atypical requirements. In sentiment analysis, they excel when compared to existing methods with the only flaws being the analysis in low-resource languages and the computational cost of training them and executing them which can restrict their usage [7]. In the Natural Language Generation tasks, LLMs show a remarkable performance with the most significant functionalities being Question answering, a task in which LLMs showcase phenomenal capabilities and also dialogue and translation. While their dialogue capacities are very good, when they are used for translation, they generally surpass all other Machine translation systems available for commercial use which is a very interesting fact considering that they were not trained for translation purposes. There is still some potential for growth, such as enhancing them in the translation from English to other languages [8]. Last but not the least, as far as multilingual tasks are concerned, there is room for improvement in the text generation field of non-Latin and poor linguistic resource languages in which LLMs performance is inadequate.

In the field of code generation LLMs display astounding capabilities with the most notable tasks being greedy search, existing code understanding, and new code generation based on instructions, dynamic programming and even code output simulation in specific languages. Some of their drawbacks in this field, are their weak management of code optimization tasks, handling data structures and recognizing code vulnerabilities [9].

Apart from general text generation tasks, LLMs can also be implemented in medical applications such as medical queries answering, either for doctors, or for patients providing factual information and even in medical care as doctor assistants. Based on research summarized by Chang et al. [10], LLMs and more specifically ChatGPT has answered correctly medical queries in several fields such as biomedicine, and genetics proving that LLMs can be useful in the specific domain, but they should be used with caution as they can easily refer wrong information in addition to facing challenges in reference citation. Another domain LLMs are evaluated on is medical examination assessment, in which they hold promising prospects as their performance is continuously improved even though this is not a field they are trained specifically for. It is concluded that in medical education and clinical routines,

they can prove to be quite useful for both students and doctors. It is even observed that LLMs provide answers that are more accurate and contextually reasoned than Google search does [11]. Lastly, when considering LLMs for medical assistants their capabilities in education and training for clinical and surgical tasks should not be overlooked. While they face several challenges with the most significant ones being misdiagnosis and patient privacy issues since medical information and images from patients must be uploaded for assessment, they showcase great potential.

In general, LLMs have emersed in people's lives quite recently and they are widely used for all the tasks mentioned above but more importantly they are being incorporated into their professional lives since more and more companies and even small businesses are openly utilizing them to optimize the work practices. Thus, since they are progressively becoming significant parts of people's lifestyles, there are several risks that should be taken into account. One of the most important threats of LLMs is their toxicity and bias on certain demographic groups along with the fact that they may generate unethical content, can all contribute to a negative impact on society. Furthermore, sharing personal information should be minimized as the models might reveal them, raising serious concerns about their utilization in sensitive areas especially in the medical sector. They can also be subject to adversarial attacks, and they have some system security issues especially in vision-language models and adversarial text prompts. Lastly, their reliability can diminish when they face questioning, negation, or misleading cues, indicating a need for more robust prompting methods to maintain consistency.

2.3 Deep learning applications in medicine and biology

As deep learning algorithms continue to be implemented in several fields, it is only natural that researchers implement them in the field of biomedicine. As previously mentioned, LLMs have already been tested for their benefits in the specific area, and they have great prospects for being utilized as doctor assistants or for learning purposes of medical students and personnel. Instead of a general application in the medical field, latest research aims to detect and treat specific diseases using Neural Network algorithms. In the following subchapter, the main Deep Learning algorithms used specifically for the medical field are going to be mentioned as well as the specific architectures implemented for each disease or condition.

This survey will only include work submitted up to the year 2022 since the most comprehensive reviews, method comparisons and thorough surveys incorporate research of earlier years.

2.3.1 CNNs

Convolutional Neural Networks (CNNs) is a Deep Learning architecture that is generally used for image classification and detection in a wide range of fields. Their advantages include their low computational cost when compared with other architectures used in medicine as well as their ability to acquire images of high quality. Two types of image representations are used as input to CNN models, 2D and 3D with the latter being computationally more expensive since the number of convolutions performed is respective to the number of dimensions of the input image. In some cases, for example in the work of Zhang Y et al. [12], 2D and 3D CNNs have been used together so that the computational cost will be lower than solely using 3D convolutions. They also state that the proposed model preserves a high accuracy in the task of the segmentation of the pancreas and this architecture could be used in several cases, most importantly for detection of tumors. Apart from the mix of 2D and 3D CNNS, a novel approach is also being used for volumetric data, 2.5D CNNs which combine the lower computational cost and memory consumption of 2D convolutions while they can still capture the most important features from different perspectives of the volumetric data by processing slices from multiple planes of the 3D scans. They are widely used in medicine for segmentation of medical scans, for the diagnosis of diseases or conditions based on features extracted from 3D medical scans in addition to the detection of abnormalities or structures in medical 3D data.

The work of Titus J. Brinker et al. [13] proved that this specific neural network architecture not only has a place in the medical field, but it is able to outperform medical specialists. Their work confirmed that by using CNNs with input images of suspect skin lesions, their ability to classify them surpassed even dermatologists' expertise. Last but not least, several popular models for image classification have been studied as for their competence in the medical field, one of which being inceptionV3 by Szegedy C et al. [14] used in X-Ray image data along with some feature extraction techniques for the identification of COVID-19 caused pneumonia [15].

2.3.2 Semantic Image Segmentation

Semantic segmentation, a widely used computer vision task in the biomedical field, entails dividing an image into segments and assigning a specific class to each segment or pixel. The objective is to categorize each pixel in the image based on a predefined set of classes. Several medical models attempting to classify each pixel in an image have been implemented in medical image analysis [16] over the years such as Full convolutional networks and adversarial training architectures, but the most popular architectures are encoder-decoder models such as U-nets. Some of the main techniques involving U-nets are attempting to increase the feature extraction efficiency of the models. For example, Su et al. [17] introduced the MSU-Net (Multi-Scale U-Net) that proposes the utilization of different convolution sizes inside each block and then the concatenation of each scale output for the next block. This way the features inside each block will be analyzed at diverse levels. They tested their model with several medical datasets such as breast ultrasound images for detection of tissue damaged by cancer and skin lesion images for skin cancer detection. Another interesting technique is nnU-Net tool which is introduced by Isense et al. [18] And involves the generation of an auto-configurable U-Net with 2D, 3D U-nets and also 3D cascade U-nets that adjusts automatically its parameters making it capable of manipulating images of various organs and body parts.

Although they are widely implemented, U-nets are inefficient in the biomedical sector when manipulating objects with a variety of shapes such as organs, because of the convolutions inside the encoder-decoder architectures. Thus, attention mechanisms have been employed to resolve this issue as their chore method aims to locate the position of the object detected instead of its shape only. The combination of semantic segmentation and attention blocks is used by Ouyang et al. [19] for COVID-19 caused pneumonia and includes the implementation of two models, one taking as input images of the whole lung and the second one, images of infected lung areas. The first one learns data representations from the initial data while the second model achieves higher attention results for smaller regions.

2.3.3 GANs

Generative adversarial networks have gained popularity since computational capacity has increased in recent years. Their ability to generate images has been deemed useful for data augmentation tasks, an ability which is quite valuable in medicine because of the occasional

lack of wide datasets for specific conditions. Several research projects have benefited from GANs ability to rebuild images, for example the employment of 2 GAN models [20] for acceleration of MRI image generation to reduce patient discomfort through rebuilding the down sampled images. Another significant endeavor is the work of Pozaruk et al. [21], which aimed to improve the PET-MRI images for prostate cancer. In general PET-CT scans expose the patient to radiation, with their work they managed to produce pseudo-PET-CT scans from PET-MR scans while also achieving a high quality of images that not only reduce the need for PET-CT scans thus reducing radiation exposure in patients, but also decrease the need for biopsy.

In their study, Ghassemi et al. [22] utilize a DCGAN to generate brain MR images. Initially, the GAN is trained with the discriminator learning to distinguish between real and synthetic MR images while extracting essential features. Following this, the fully connected layers in the discriminator are substituted with a softmax layer, and the model undergoes retraining.

GANs have also been used for skin lesion classification [23] as the previous architectures have, achieving good results for COVID-19 caused pneumonia or inflammation as well, with the use of CT scans, MR scans and chest X-rays as input and lastly for data augmentation in this specific domain. Last but not least GANs are used for Alzheimer's disease classification by Zhou et al [24] with the proposal of 2D, 2.5D and 3D convolutions for handling and improving MR images of the brain which are then imported into a classifier that provides the result of the patient's possibility to suffer from Alzheimer's disease.

2.3.4 VAEs

In the medical field, image segmentation is a key application for VAEs. These models are designed to produce new images based on the original input, highlighting significant areas of interest. The resulting images include additional information to identify specific targets, such as abnormal features or differentiated regions. Although GANs are more frequently utilized in medical research than VAEs, many researchers prefer using hybrid models that integrate the best aspects of both GANs and VAEs. A hybrid model consisting of a GAN and a VAE model is proposed by Nakao et al. [25] whose objective was to detect outliers in chest radiography data. While their technique is quite precise, they state that it cannot take the place of a doctor as it has not achieved the diagnosis of such lesions.

The implementation of VAE architectures solely, without their combination with GANs, is also observed in several papers. Such work is performed by Marimont and Tarroni [26] who study the VQ-VAE model using MR images of the brain and abdomen. This model maps features in the latent space to a dictionary of keys, serving as an embedding. They found that their unsupervised anomaly detection method outperformed standard VAEs, especially with brain images, likely due to the higher variance in the abdominal data.

2.3.5 Conclusion

In conclusion, there is an abundance of methods and techniques used in the biomedical sector with each one of them utilizing different architectures. It has to be noted that some small hyper-parameter adjustments when performed correctly, can lead to phenomenally more accurate results in a plethora of architectures. This proves that the expertise in Deep learning in the medical field is more important than the use of different deep learning architectures.

Moreover, the problems doctors are facing are the lack of data especially for rare diseases, the amount of time it takes to label each specific image and the fact that only experienced staff can label the data. These problems can be resolved by semantic segmentation, data augmentation and several other techniques implemented in Neural Networks. Although there is great progress, several steps still need to be made to ultimately resolve the issues above such as the development of more advanced unsupervised learning methods, the creation of larger and more diverse medical image datasets, and improvements in transfer learning techniques to leverage pre-trained models for better accuracy with limited data.

As Jan Egger et al. [27] state, a serious challenge faced in the medical field, is the very small incorporation of Deep learning into clinical workflows. Furthermore, their survey concludes that while there are a lot of variations of techniques used in biomedical data, the main objective of researchers is to outperform the state-of-the-art methods already created without taking into consideration their ability of implementation in actual medical procedures, or the diagnosis of human data aiding doctors and medical staff since these methods are usually restricted to “ideal scenarios” [27].

Chapter 3

Pruning Deep Neural Networks

3.1 Introduction to Pruning

As evidenced by the analysis of Deep Learning in the previous chapter, it can be concluded that Large-scale Neural Networks are utilized progressively more often. The expansion of Deep learning across various fields has made it important for researchers and companies that build such applications to achieve high accuracies in deeper and deeper models. Networks of such volume require an extremely high computational capacity and vastly powerful GPUs, which is something that very few possess especially in the academic community excluding a significant percentage of people from being able to create or even test such extensive networks.

In several surveys and research papers it is stated that a large number of the trainable parameters is not essential to the training process and even though a model can reach a high accuracy, a substantial percentage of its parameters are in fact useless. As stated by Gromov, Andrey, et al. [28] especially in Large Language Models a tremendous percentage of them is not needed. They also note that approximately half the layers inside deep models can be deducted without affecting the model's performance.

Pruning has been widely popular in recent years as it is the practice of removal of unnecessary parts of the model such as weights, nodes, channels or even whole layers from Neural Networks to decrease their inference time, their Floating Point Operations (FLOPs) during training and the memory they require. Pruning has proven to be quite effective, as it manages to significantly diminish the computational and time cost of models while still preserving and in some cases increasing the accuracy of the model after pruning. This happens as pruning

can mitigate overfitting and help the model generalize on unseen data. Furthermore, pruning makes it easier for deep learning models to be deployed in resource-constrained devices such as phones and embedded systems as well as lower the power consumption leading to extended battery life in such devices.

In this chapter the main pruning methods and techniques are going to be analyzed as well as the most significant algorithms created for pruning Neural Networks.

For the analysis of various pruning methods, a handful of surveys have been studied but the most recent and useful ones are deemed to be [29] (23) and [30] which offer valuable insight.

3.2 Types of Pruning methods

3.2.1 Structured and unstructured pruning

Structured and unstructured pruning are two of the main pruning categories. Structured pruning involves the overall deduction of entire layer structures such as norms, filters, channels or even layers. Structured pruning is efficient since it does not require hardware with high computational capacity for the compression of the model. On the other hand, unstructured pruning is a method that requires acceleration via specific hardware and libraries that aims to prune specific weights or parameters inside a layer creating a sparse architecture that not all GPUs can handle. Apart from the methods above, semi-structured pruning is proposed by several research papers that attempts to combine the fine characteristics from each one of the methods above which are, the structural regularity making it possible for regular hardware to implement, as well as a high precision [29].

3.2.2 Pruning Pipelines

While pruning is mainly discerned into structured and unstructured pruning, there are lots of ways to categorize pruning techniques [31]. One of them is by the point in the training process of the model during which the structure is being pruned. In other words, the possible pruning pipelines of a model for static pruning as mentioned in the work of Cheng and Zhang et al. [29] fall into the following categories, Pruning Before Training (PBT), Pruning During Training (PDT) and Pruning After Training (PAT). The first one, PBT involves the random

initialization of a network and the pruning of weights based on a certain criterion before first training it. As mentioned in their work [29], during PBT the model is first pruned and then since it is a sparse network, is trained with a fixed sparse pattern until it converges. The main goal of the specific method is to benefit from the low time needed as the model is trained once after the pruning has been executed. PDT also uses a randomly initialized model as the previous method did, only now the pruning occurs during the training process. This method is not widely implemented because of its intricate nature stemming from the complex dynamic operations that need to be performed. Last but not least, PAT is a procedure that involves pruning after the model is first trained and then finetuning or retraining the model until convergence.

It is widely thought to be the most useful out of the three since the filters and layers of the model can be more efficiently pruned with the relationships between them being thoroughly captured after the network is pretrained. The previous pruning pipelines involved only static pruning, while Runtime pruning, or dynamic pruning refers to the process of pruning neural network weights dynamically based on each input of the network. Pruning occurs during inference rather than during training or in a post-training step. This technique adapts the model's complexity during training, in regard to the input data and the current state of the model.

3.3 Structured pruning

Structured pruning will be analyzed to a large extent since the objective of this work is the presentation of a novel structured pruning technique. Structured pruning can be sorted into several categories. It is important to note that the criteria for structured pruning may coincide, creating in some cases hybrid techniques that utilize several pruning criteria for higher accuracy and efficiency.

3.3.1 Weight-dependent criteria

A simple technique is pruning based on Weight-dependent criteria. This general technique involves pruning based on the magnitude of the weights inside each filter. It can be divided into two categories, filter norm and filter correlation based criteria. Pruning based on filter norm suggests measuring the filter norms either l_1 or l_2 and using one of these as the metric

for pruning. If the norm value is small, then the filter is deemed weak, and it is pruned. This specific subcategory only uses the magnitude of the norms as a metric while the second subcategory of weight-dependent techniques, filter correlation, utilizes the relationship between several filters to conclude if they are important or not. A popular example of filter correlation based pruning is Filter Pruning via Geometric Median (FPGM) [32], this technique involves the layer-wise comparison of filters based on their distance from the geometric median. The filters closer to the median are pruned as they seem to learn common features from the input image which have already been represented from other filters of the same layer.

3.3.2 Activation Based Methods

Benefiting from the model's activation maps, activation-based techniques can be subdivided into three categories based on which layer's activation maps they are using for pruning, as mentioned in the work of He and Xiao [30]. These are, "pruning techniques based on the current layer, adjacent layers and all layers" [30]. A characteristic example of activation-based techniques that utilize activation maps from the "current layer" is Channel Pruning (CP) [33] which uses a layer's activation maps to guide the pruning of its filters. It minimizes the reconstruction error of sparse activation maps via two steps: (1) identifying channels to prune by solving the LASSO regression with fixed weights, and (2) fine-tuning the weights to minimize reconstruction error after pruning. Pruning considering Adjacent layer activation maps is performed by Ding et al. in [34] who introduce Approximated Oracle Filter Pruning (AOFP) a technique consisted of three stages, the first one involves "damage isolation" where the network's damage is being isolated into the current (l) and next layer ($l+1$) so that the damage is unnoticed by the 3$^{\text{rd}}$, in line layer ($l+2$) . In the second stage AOFP utilizes a multi-path framework for parallel scoring and fine-tuning and thirdly, it applies a binary filter search method to resolve issues within this framework.

Lastly a very interesting approach that exploits multi-layer activation maps is Neuron Importance Score Propagation (NISP) [35], the idea behind which is that if a neuron contributes slightly to the model's "Final Response Layer (FRL)" [35], then it is highly likely that the specific neuron is redundant to the whole model. The procedure begins with the calculation of the feature score based on any feature ranking method (FRL). Then this score is passed back through the network to earlier layers and neurons with low importance scores in each layer are subsequently pruned, a process which prevents them from influencing the

importance scores of neurons in prior layers.

3.3.3 Regularization based techniques

Batch Normalization (BN) layers are usually added to deep CNN models for a better generalization [30]. The procedure followed during BN is the following, during training, BN calculates the mean and variance for each activation within a mini-batch of data. It then transforms the activations to achieve a zero mean and unit variance. This transformation is learned for each layer and channel, enabling the network to adjust to the specific properties of the data it processes. Applying a sparsity regularizer to BN parameters in networks with batch normalization layers helps in determining which channels or filters to prune, thereby achieving structured sparsity and simplifying the network. This technique falls into the subcategory of Regularization based techniques, called Regularization on BN Parameters.

In some cases, BN is achieved without adding BN layers to the model, simply by using additional parameters called learnable gates. Since these gates are acting as controllers to determine which parts of the network should be kept or removed, networks can guide the pruning process effectively without relying on BN layers. This specific method is pruning based on the regularization of extra parameter. Convolutional Re-parameterization and Gradient Resetting (ResRep) [36] is a very interesting instance of regularization based on extra parameter as it involves restructuring the CNN into two sections. The first section, known as the “remembering part”, is responsible for maintaining the model’s performance and is not subjected to pruning. Then the “forgetting part” adds 1x1 convolutional layers, called compactors, after the batch normalization layers. During training, a modified Stochastic Gradient Descent (SGD) update rule is applied exclusively to these compactors allowing them to be pruned (“forget”), while the main convolutional layers remain untouched to preserve the model’s overall performance.

The last subcategory of Regularization based techniques, is regularization based on filters, used to achieve structured sparsity by directly targeting the filters within the convolutional layers. An example of this technique is the work of Wang et al. [37] that introduces Growing Regularization (GREG) an algorithm that implements two distinct algorithms. The first algorithm takes care of establishing a pruning schedule by employing the l_1-norm to create a pruning mask and then it uses a gradually intensifying l_2 penalty to diminish their weights to zero over time. The second one applies an escalating regularization to take advantage of

the Hessian information and overall, this technique uses a progressively increasing penalty to decide which filters to prune.

3.3.4 Optimization techniques

In the context of Neural Network pruning, optimization techniques include several methods used to identify redundant filters, neurons or channels and then remove them, while maintaining structural sparsity in the model. Some common "Optimization tools" [30] as it is mentioned in the work of He and Xiao are Taylor Expansion, Variational Bayesian Methods and SGD-based Methods. Taylor Expansion [38] is a technique that is used to approximate the change in the loss function when a filter is removed in order to identify the importance of each filter. Filters are removed from the network and those that present a smaller increase in the loss function, are considered less important and are subsequently pruned.

Variational Bayesian Methods [39], generally use probabilistic models to determine the importance of filters through the analysis of the "prior and posterior" distributions of the model's parameters and the filters chosen for pruning are those who have lower importance scores. Finally, SGD-based Methods, adjust the gradient update rules to encourage sparsity. There are various modified versions of SGD utilized for pruning which are used to update the network's weights while they identify and remove redundant filters.

3.3.5 Dynamic pruning techniques

Dynamic pruning is another pruning method which involves adjusting the structure of a neural network in an adaptive manner. Dynamic pruning can occur either during training or during inference phases.

The Dynamic pruning during training, or soft pruning, aims to maintain the model's ability to represent data effectively using a flexible pruning mask. which is modified throughout the training process, allowing the model to recover from potentially poor pruning decisions.

Dynamic pruning during inference on the other hand changes the network's structure based on the complexity of the input data. The idea behind this method is that a simple image with relatively clear does not require a high number of resources especially when it is compared to the resources required for a more complex and detailed image [40]. This approach helps retain balance between resource utilization and model accuracy, ensuring the network operates efficiently according to the demands of different input samples.

The work of Chouliaras et al. [41] belongs to dynamic pruning in the training phase category as the sparsity level of the model is altered during training rather than deciding on the sparsity level beforehand or adjusting it based on the complexity of the input data during inference. It involves pruning or restoring the connections between layers of feed forward neural networks based on three proposed rules,

Linear Decreasing Variation (LDV) rule or Oscillating Variation (OSV) rule or Exponential Decay (EXD) [41]. The goal of this technique is to sparsify a Multi-Layer Perceptron Network.

3.3.6 Additional pruning Techniques

Another type of structured pruning is network sparsification based on structured sparse topologies. In this category, node connections are removed or reinstated based on structures or patterns. A very interesting approach was provided by Fragkou et al. [42], this work proposes five algorithms, alternatives of a specific concept which includes the reduction of model parameters through the creation of structured sparse topologies. This is accomplished through the utilization of smart link rewiring methods in order to create a structured topology from another structured topology. Out of the five algorithms created, their most successful method

SF2SFrand [42] begins from scale-free topologies and creates scale-free-like topologies significantly reducing the model's parameters.

3.4 Limitations of Pruning Techniques

The techniques presented above, all have important advantages but in general do not combine the attributes below:

- Enhance the model's accuracy post-pruning
- Maintain the model's adaptability to operate efficiently across diverse datasets
- Minimize the model's loss values
- Ensure pruning decisions require minimal computational cost

More specifically, while effective in reducing the trainable parameters and improving the computational efficiency of a model, structured pruning often aggravates the accuracy of the

model. This is due to the fact that most structured techniques if not all, perform aggressive pruning of entire layers or channels paying attention only to lowering the model's inference time and memory footprint and conclude with a weaker model that has lower accuracy and higher loss values. Furthermore, even though the pruned models created by these techniques can be implemented in resource-constrained devices, the pruning methods are usually computational costly to the point where the need for a highly capable GPU is imperative. The former techniques deprive models of flexibility, since they are not able to adapt and learn varying image attributes once they are pruned in a way that filter importance is not taken into account and blocks of the models are pruned carelessly. Thus, the formation of a pruning technique that cautiously prunes unwanted model parameters taking into consideration both the model's accuracy as well as the reduction of its size, is crucial.

Chapter 4

HaDiPCS Pruning Technique

In this chapter a novel pruning technique Hausdorff Distance based Pruning for Convolutional Neural Networks (HaDiPCS) will be presented which utilizes the Hausdorff Distance to calculate dissimilarity values between filters to conclude which filters in the model should be pruned. It is a weight-dependent, similarity-based pruning technique that belongs to the category of Pruning After Training (PAT) methods and implements a layer-wise filter importance assessment and pruning. This chapter begins with a mathematical analysis of the Hausdorff Distance (HD) calculation, continues with the explanation of how this distance is utilized in the HaDiPCS pruning technique and lastly the overall technique is thoroughly defined and analyzed.

4.1 Hausdorff Distance explanation

The Hausdorff distance, which is the core mechanism of the specific method, has been mostly used in computer vision tasks, pattern recognition and shape analysis and especially in the medical field where it is used for various tasks, one of which being the evaluation of medical image segmentation applications in MRI or CT scans [43]. It computes the dissimilarity between two sets of data, and it can be used as well for calculating the level of similarity between two images [44].

For a full understanding of the technique someone must know the directed Hausdorff distance and the maximum Hausdorff distance. Both equations for the Hausdorff Distance explanation and mathematical analysis were derived from [43]. The directed Hausdorff distance H_d finds the maximum distance between a point a in a set A of data and the nearest

point b that belongs to another set of data B:

$$H_d(A,B) = \max_{\forall a \in A}(\min_{\forall b \in B} ||a,b||) \tag{4.1}$$

Where $||a,b||$ is the norm between a and b using any base distances such as the Euclidean distance, the Manhattan distance or even the Chebyshev distance functions.

$$H_d(A,B) \neq H_d(B,A) \tag{4.2}$$

which means that the Hausdorff distance is not symmetric for two sets of points and always outputs a positive number. The Maximum Hausdorff distance H_m is the maximum value out of the directed Hausdorff distances of two sets of points:

$$H_m = \max(H_d(A,B), H_d(B,A)) \tag{4.3}$$

The Hausdorff distance considers the spatial position of each point inside the sets [43]. In several applications the use of the Hausdorff distance is susceptible to outliers [45, 46] and can reach a high computational complexity. In this work the Hausdorff distance is calculated using the sklearn's package directed_hausdorff, an optimized approach based on the work of Abdel Aziz Taha and Allan Hanbury [43] which calculates the directed Hausdorff distance H_d with a "nearly linear complexity" [43].

The directed Hausdorff approach is calculated instead of the max, in order to gain more insight from the model's behavior.

4.2 Hausdorff Distance in pruning

Hausdorff distance (HD) is calculated based on the weight matrix of the filters inside each layer and channel. The HD is calculated separately for each layer thus, to avoid repetition in the method analyzed below, the calculations mentioned will be referring to the same layer. The principal notion of the method is that several filters learn common features of the input image, making most of them completely redundant since they do not provide any new and valuable information to the model. The HD between 2 filters provides the knowledge of how dissimilar these filters are to each other. The lower the result of the HD calculation, the more similar the features learned by these specific filters are (lower dissimilarity).

The main algorithm entails the calculation of the distance between the weight of one filter inside a specific channel, and the weight of another filter inside the same channel. This

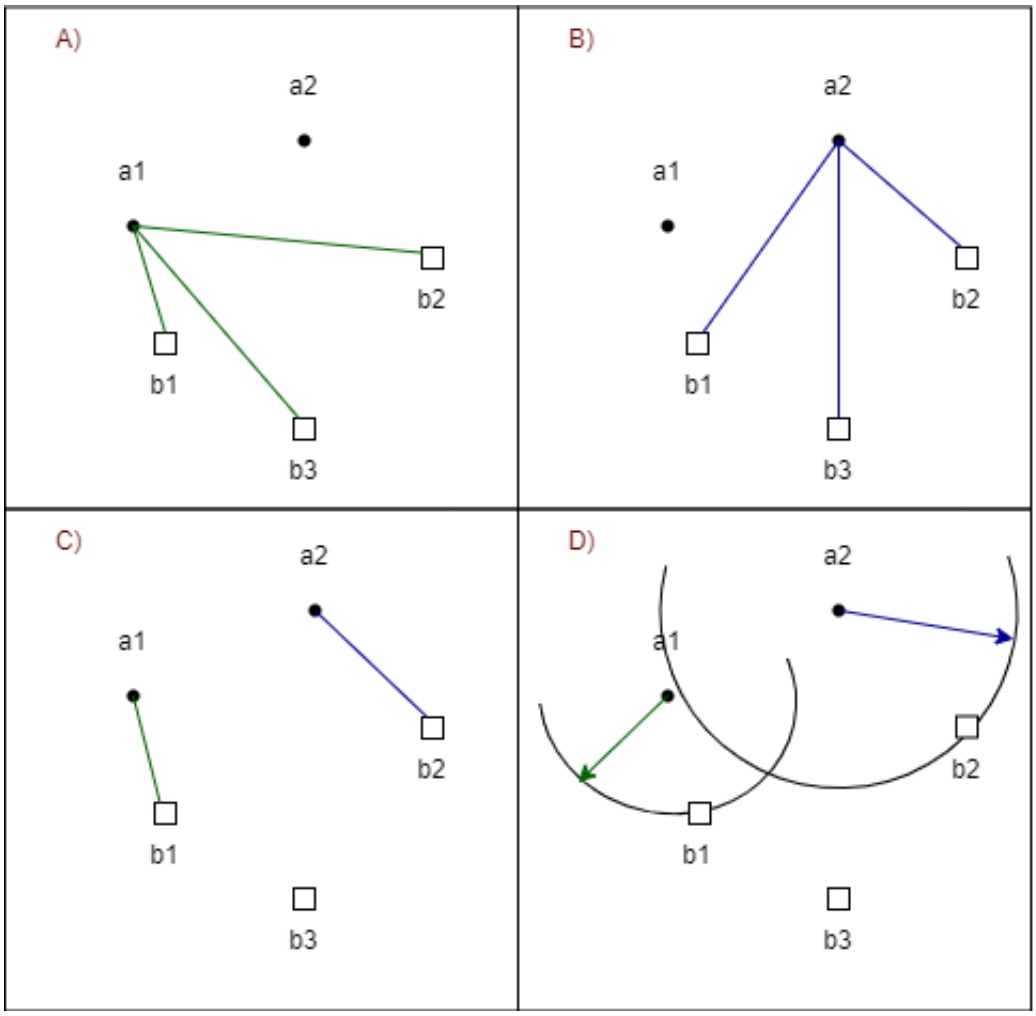


Figure 4.1: Maximum Hausdorff distance calculation between 2 sets of points A and B. A) Distances between point a1 that belongs to set A and every point from set B. B) Distances between point a2 that belongs to set A and every point from set B. C) Choice of the 2 minimum distances between points of a1 and a2 from B respectively. The maximum of these 2 distances is the maximum Hausdorff distance which in this case is the distance between a2 and b2 shown in a blue line $H_m = distance(a2, b2)$ D) Every point from set A has a maximum distance of H_m from set B

distance is calculated for the combination of all filters inside each channel of the specific layer. The only filter combination not computed for avoidance of extra calculations, is the distance between the same filters, since $H_d(A, B) = H_d(B, A)$ if $A = B$. Then the distances between the filters from different channels are summed together and divided by the total number of the channels of the layer. The value outputted from this procedure is then placed in a matrix referred to as the Hausdorff Distance matrix (HD matrix).

$$HD(F_1, F_2) = \frac{\sum_{i=1}^{i=c_l} H_d(F_1, F_2)_i}{c_l} \tag{4.4}$$

Where:

c_l = Total number of channels in a specific layer

$H_d(F_1, F_2)_i$ = The directed hausdorff distance between weights of Filter 1 and 2 inside a specific channel c_l

l = the current layer

In the first layer the number of channels the method will have to calculate is the same as the number of channels of the input image, for example a grayscale image will have 1 channel while a colored RGB image will have 3 channels. As the method proceeds in deeper layers, the number of channels is augmented.

The HD matrix created from this procedure is not symmetrical since the calculation of the directed HD is not symmetrical for two sets of points as mentioned in 3.1. a choice made in order to acquire more insight into the dissimilarities of each pair of filters. Moreover, the HD matrix has a diagonal containing zero values. If the two HDs of 2 filters $H_d(F_1, F_2)_l$ and $H_d(F_2, F_1)_l$ are significantly different from each other, then these 2 filters do not necessarily learn similar characteristics.

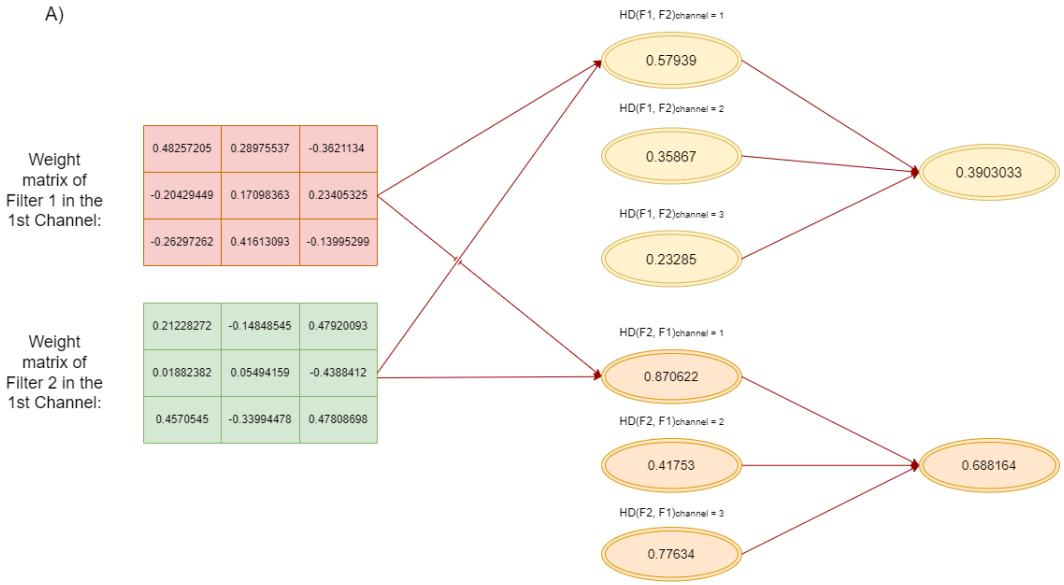
4.3 HaDiPCS Pipeline and Overall Technique

After the creation of the hausdorff matrix a threshold is set, below which the filters will be checked by the following methods for pruning. Two different ways of threshold setting are proposed, the first is the percentile based threshold in which the threshold is set at the value of the percentile of the matrix chosen by the user indicating the percentile of the dissimilarity values that will be checked for pruning.

The last one is range-based threshold:

$$Threshold = \min(HDmatrix) + a * (\max(HDmatrix) - \min(HDmatrix)$$

Where $\max(HDmatrix)$ and $\min(HDmatrix)$ are the maximum and minimum dissimilarity values of the HD matrix respectively and a is a configurable value. Range-based threshold and percentile-based threshold are the methods mostly used in this work. After the threshold



B) Resulting HD matrix

Filter	1	2	3
1	0	0.3903033	0.4418
2	0.688164	0	0.6293t
3	0.23098	0.84702	0

Figure 4.2: Indicative calculation of 2 values of the Hausdorff Matrix. A) Left: weight matrices of the first channel inside the first layer of the model with 3 input channels, for filters 1 and 2. Center: corresponding directed Hausdorff Distances (HD) are calculated from the weight matrices, directed Hausdorff Distances for channels 2 and 3 of the specific layer are shown as well. Right: Based on equation 4.4 the values for $HD(F_1, F_2)_1$ and $HD(F_2, F_1)_1$ are calculated and their results inserted in the positions (0,1) and (1,0) of the matrix respectively. B) Resulting Hausdorff Distance matrix for 3 filters and 3 channels in the first layer of a model

is set, the algorithm finds the pairs of filters that have a dissimilarity value below this threshold. If both pairs of the same filters have a value lower than the threshold, they are marked as pairs to be pruned for example for $HD(F_1, F_2)$ and $HD(F_2, F_1)$, filters 1 and 2 are marked as filters with low dissimilarity.

4.3.1 Example of the method

In the Resulting HD matrix in the figure 4.2 if the threshold is set at 0.5, the filters that will be marked for pruning, are filters 1 and 3 as, based on the matrix positions (0,2) and (2,0) both have dissimilarity values lower than the threshold thus filters 1 and 3 learn common image attributes. These filters will be examined by the following methods which will determine which one is going to be pruned.

There have been 2 methods developed for filter pruning based on their dissimilarity, these are 3-pair pruning and 2-pair pruning.

4.3.2 3-pair pruning

The first method checks for 3 pairs of filters from the list above (each pair should have both dissimilarity values of the HD matrix below the threshold) that come from the mix of the same filters, for example pairs (F_1, F_2), (F_2, F_3) and (F_3, F_1). If these pairs all have dissimilarity values below the threshold, then two out of the three filters are randomly chosen for pruning and added to the pruning list. The remaining filter for each comparison is then marked as important since at least one of the three filters must remain so that the image features they commonly learn are definitely represented.

4.3.3 2-pair pruning

This method performs comparisons in the same way, but this time the dissimilarity values of 2 pairs of filters have to be lower than the threshold. Again, one of them is randomly chosen for pruning and the other one is stored as an important filter that should not be pruned.

4.3.4 Combination of 3-pair and 2-pair pruning

In this work, both methods are utilized sequentially for obtaining better results. The first to be implemented in HaDiPCS is 3-pair pruning which ends up selecting more filters for pruning in higher threshold values. In lower threshold values this method results with fewer filters for pruning since there is a lower probability for several combinations of 3 filters to have dissimilarity values below this threshold. During this procedure, the filters that were checked for pruning but not chosen, are inserted into a list and marked as important. The remaining pairs of filters that are not contained in neither list mentioned above, are examined

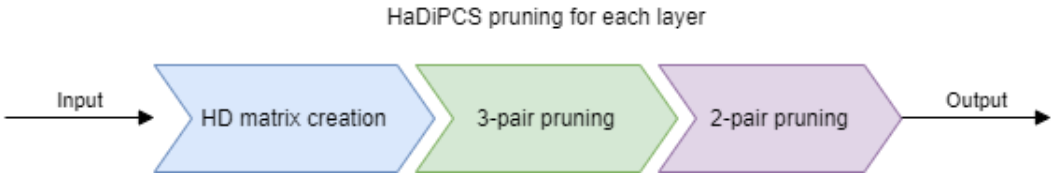


Figure 4.3: Order of pruning methods of HaDiPCS technique for the pruning of a specific layer

by the 2-pair pruning method. If the threshold is high, then the 2-pair pruning method does not conclude with any filters for pruning since the majority of them are chosen by the former method either for pruning or to be kept. When the threshold is lower, this method efficiently complements 3-pair pruning concluding in a comprehensive pruning technique.

4.3.5 Overall technique

The HaDiPCS pruning technique is implemented for each convolutional layer of the model separately. The threshold can be set at a different value for each layer depending on its depth inside the model and the values of its hausdorff matrix. The pruning process is as follows, beginning with the first layer selected for pruning, a new HD matrix is created based on the values of the layer's weight, the 3-pair pruning and 2-pair pruning processes are executed in the specified order and then a sorted list of the filters chosen for pruning is outputted. The filter pruning is performed by deleting the filters of the layer using a custom Keras surgeon function. This process is repeated for all the convolutional layers of the model and whenever there is a different type of layer in the model such as max pooling, it is replaced by a copy of the original layer with input values set as the output values of the previous convolutional layer pruned. This process is repeated for all layers selected and then the pruned model is fine-tuned. The layers chosen for pruning will be mentioned in chapter 4 as well as the reasons for their selection.

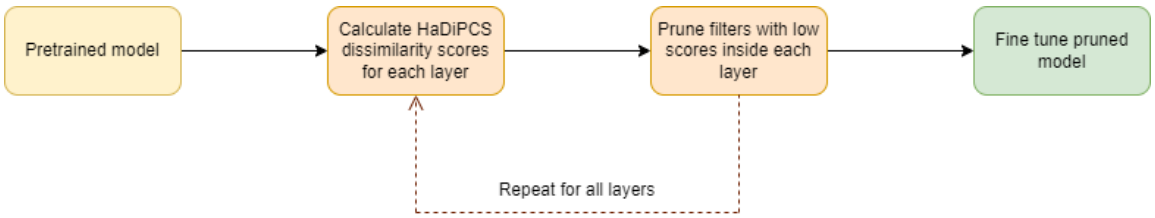


Figure 4.4: Pipeline of HaDiPCS technique for the pruning of a model

Chapter 5

Experimental Evaluation

5.1 Datasets, Model and Preliminary Analysis

5.1.1 Datasets and Model Architecture

The network chosen for the assessment of the method is a custom architecture shown in figure 5.1, which was chosen because of its high accuracy of 0.9909 in the MNIST dataset and its mediocre to low accuracy of 0.6787 in the CIFAR-10 dataset while it can also be trained with low computational and time cost without the need for a highly capable GPU. The CIFAR-10, a colored RGB dataset with 3 input channels and the MNIST dataset which contains Grayscale images were chosen for the execution of the HaDiPCS method with various threshold values as this will offer a comprehensive review of the technique for both colored and grayscale images. Furthermore, after the implementation of the pruning technique, it can be concluded if the method apart from deducting filters, it maintains high accuracy for the already accurate model and if it can increase the accuracy of the model in the CIFAR-10 dataset in which it shows a weak performance.

The following datasets were utilized for the evaluation of the pruning technique, SVHN [47] a dataset for digit recognition obtained from house numbers in Google Street View images, Fashion-MNIST [48] which contains 70,000 grayscale images of 10 fashion categories, and datasets mentioned above MNIST [49] which also contains 70,000 images of handwritten digits and 10 classes (digits 0-9) and CIFAR-10 [50] that contains 60,000 colored images with 10 classes of various animals and vehicles. Lastly, MNIST-C [51] was utilized, a corrupted version of MNIST that was created to test a model's robustness by adding several

types of noise in the initial images.

The model architecture utilized for the assessment of HaDiPCS is shown in the image 5.1 and it is comprised of 6 convolutional layers that contain various numbers of filters with the minimum and maximum being 32 and 128 respectively, 3 max pooling layers, 1 flatten and 3 dense layers

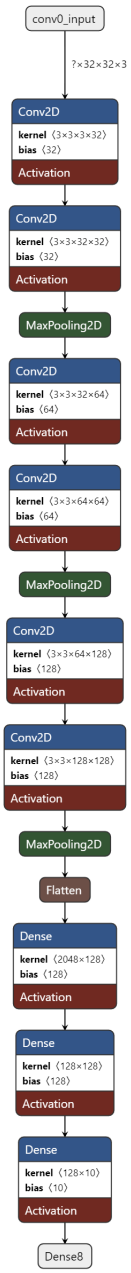


Figure 5.1: Model Architecture

5.1.2 Evaluation Metrics

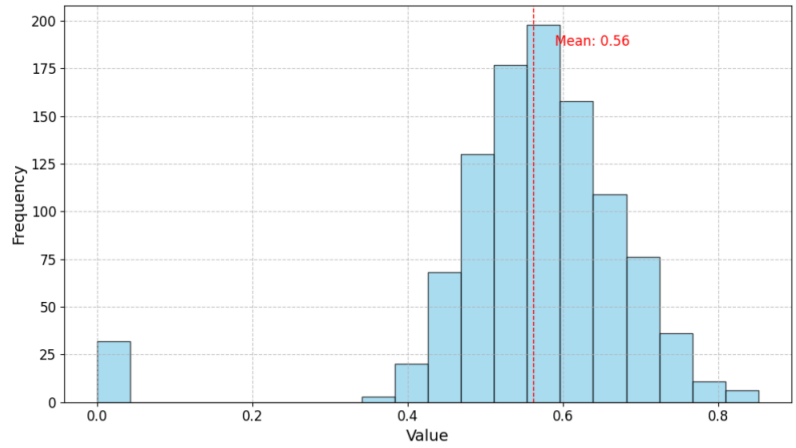
The metrics that were studied in order to assess HaDiPCS are:

- Accuracy: The model's accuracy post-pruning is compared to its accuracy before pruning to understand if the model's capability to predict data has enhanced or worsened
- Loss: The baseline model's and pruned model's loss difference is contemplated to conclude whether the pruned model's or baseline model's predictions are closer to the actual outcomes or not
- Inference Time: The pruned models are assessed as to the change they cause in the inference time of the model, which refers to the time it takes a model to make predictions on unseen data. In general faster inference times are desirable since the models process data quicker.
- Trainable Parameters: A reduction in the mode's trainable parameters is closely connected to the reduction of the model's memory footprint and inference time, both really important for real-time applications
- FLOPs: Floating Point Operations or FLOPs is the amount of operations that are performed in this case during the inference phase. It serves as a measure of computational complexity, indicating how much computational work is required to process data through the neural network.

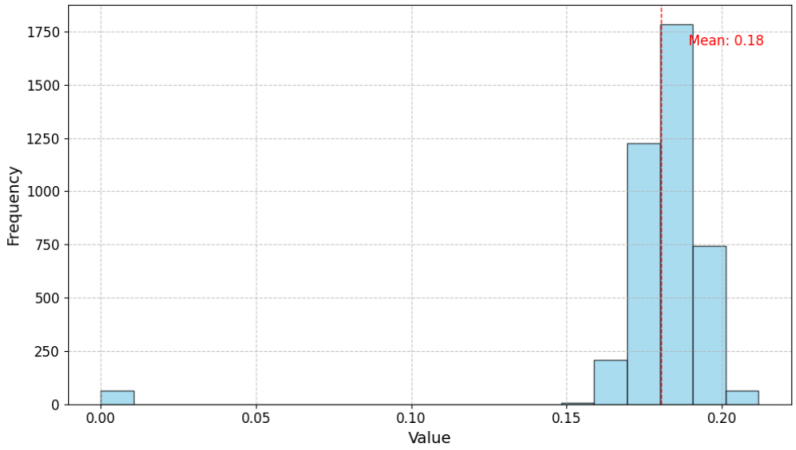
5.1.3 Annotations before pruning results

Before presenting the result of the experiments, some explanations must be provided as to which layers were chosen for pruning. More specifically, in the experiments where the first convolutional layer of the model was pruned, the accuracy of the model dropped significantly even when the threshold was set at fairly low values. Therefore, the first layer is not pruned in the experiments presented below.

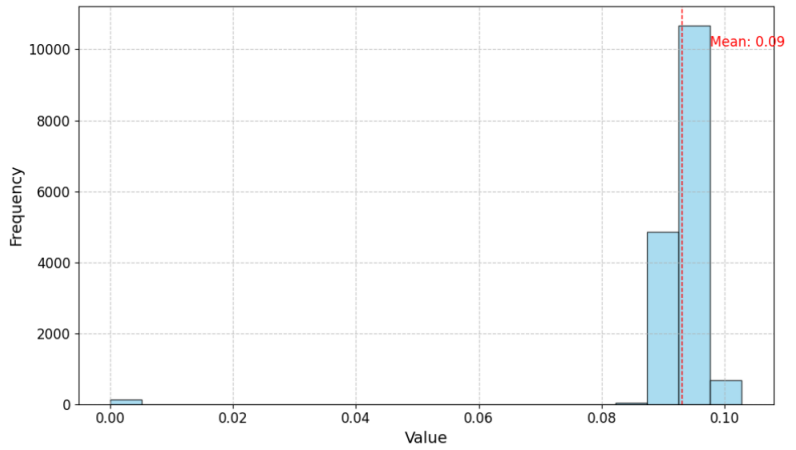
As evidenced by the figure 5.2 a), the majority of the dissimilarity values in the first convolutional layer are above 0.5 which means that the filters in the first layer in general learn different image features and thus they are needed for the proper performance of the model. The dissimilarity values of the next layer are remarkably lower exhibiting the imperative



a) Dissimilarity Values from the Hausdorff matrix in the first Convolutional Layer of the model



b) Dissimilarity Values from the Hausdorff matrix in the second Convolutional Layer of the model



c) Dissimilarity Values from the Hausdorff matrix in the last Convolutional Layer of the model

Figure 5.2: Distributions of dissimilarity values from Hausdorff Distance Matrices in CIFAR-10 dataset in the first, second and last convolutional layers of the model.

need for pruning. At the last layer, the dissimilarity values are very close to 0 with a notable mean value of 0.09. This means that only a small number of the filters in the deeper layers

are performing worthwhile tasks and the rest of them need to be deducted from the model. The high accumulation of 0 values is due to the null diagonal of the Hausdorff Matrix. It has to be noted that the values of the diagonal are not taken into consideration when calculating the pruning threshold of each layer.

5.2 CIFAR-10 Evaluation Results

After executing the HaDiPCS technique using various threshold values in the CIFAR-10 dataset, several pruned models were created and then retrained for a few epochs. As portrayed in figure 5.3, which represents the pruned models' accuracy and loss for each threshold value, the HaDiPCS increases the accuracy of the pruned models in half and the accuracy across all pruned models does not drop below 0.66 which is a slightly smaller accuracy from the baseline model. Furthermore, the loss of the model is steady, and it has even decreased in some cases.

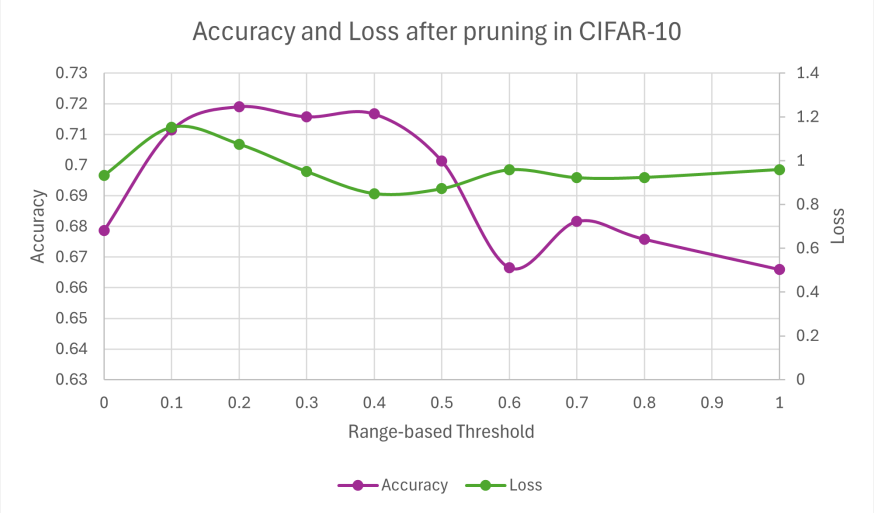


Figure 5.3: Accuracy and Loss in CIFAR-10 dataset that occurred after pruning the model with different range-based thresholds that spread from a minimum value of 0 representing the baseline model, and then each point in the chart is a pruned model with a different threshold value (different number of dissimilarity values are taken into account for pruning) with the highest threshold value being 1 which indicates that all dissimilarity values were taken into consideration for pruning.

It is worth noting that in the case of the 0.6 threshold, the parameters as well as the FLOPs of the model are significantly lower than the respective values of the model pruned with a

0.7 threshold. A mention should also be made about the behavior of the pruned models at thresholds higher than 0.5, in which the FLOPs do not present a considerable reduction and the model parameters even increased in the case of the 0.7 threshold compared to the threshold of 0.6.

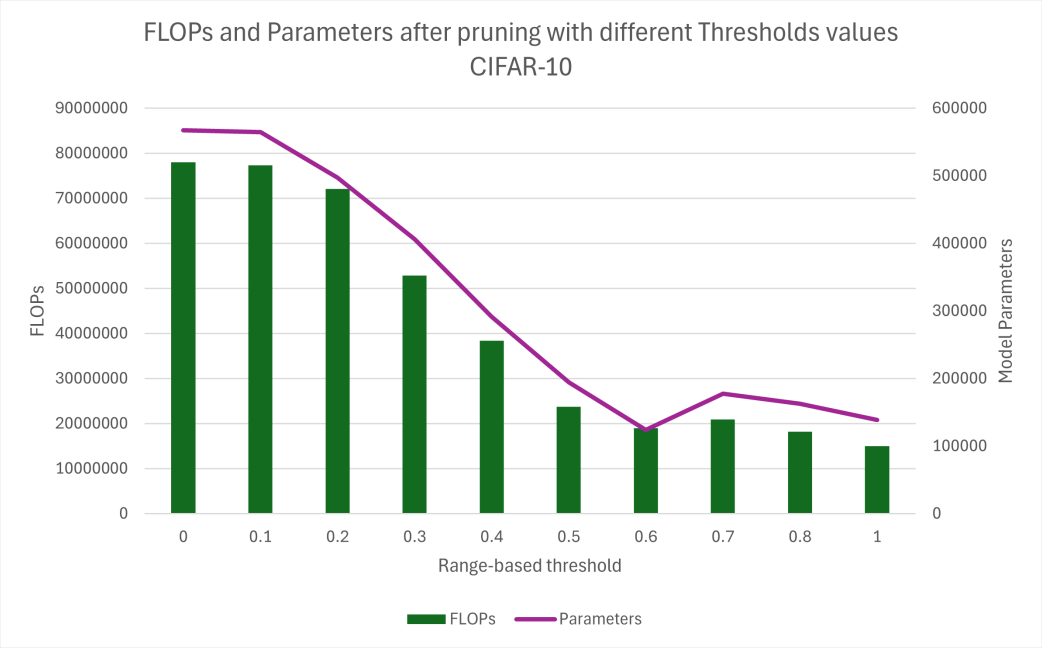


Figure 5.4: FLOPs and trainable parameters that result from pruning the model with different range-based threshold values in CIFAR-10 dataset

This happens because while the threshold is set, it does not indicate the pruning percentage, it only indicates the values that will be checked for pruning. In these cases, the HaDiPCS checks a large amount of dissimilarity values, and it can only prune a specific portion of them due to the mechanisms of the 3-pair and 2-pair pruning methods it implements. If the method has already chosen this number of filters for pruning, approaching the maximum filter volume that can be deducted without deteriorating the model's behavior, several unnecessary iterations will be made for the remaining dissimilarity values. More specifically, the algorithm will continue checking if filters are chosen for pruning even though most of them have already been selected either for pruning or been marked as important to stay in the model, in order for the image features commonly learned to be represented by at least 1 of the remaining filters.

As for the inference time of the model, it is decreasing as the threshold value increases which is logical seeing that the trainable parameters and FLOPs of the respective pruned models minimized. The inference time of the baseline model is 2 times higher than the in-

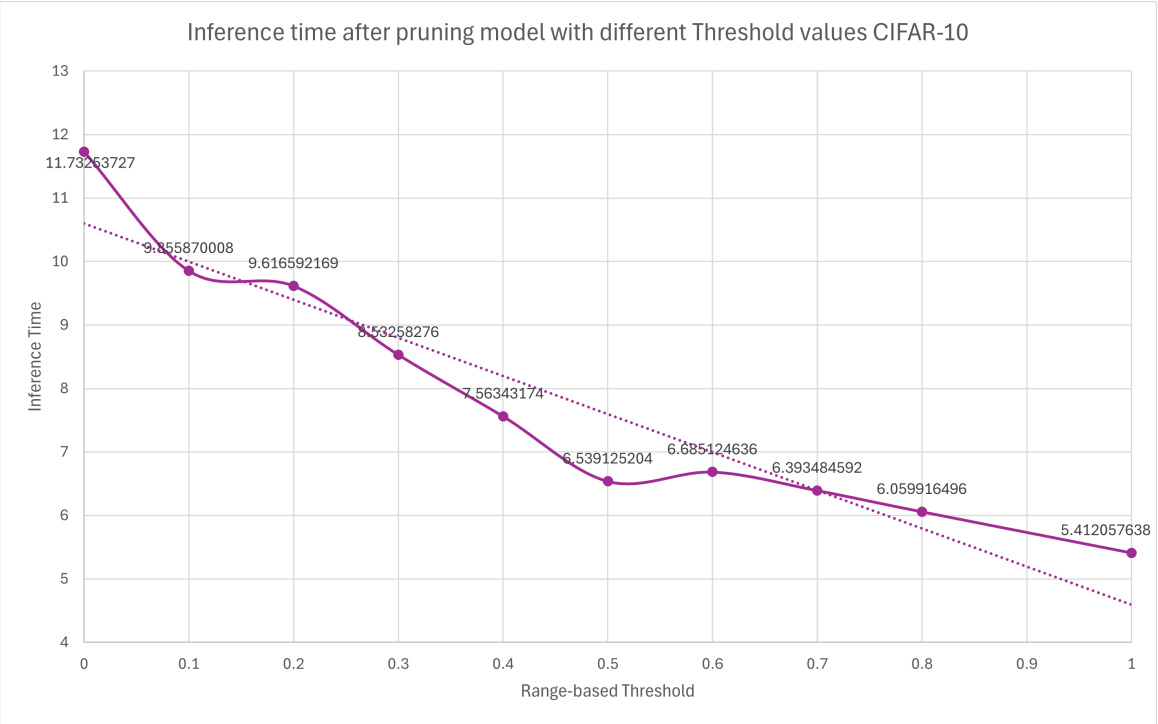


Figure 5.5: Inference time of models after pruning with various range-based threshold values in CIFAR-10 dataset

ference time of the pruned model with threshold number 1, for the pruning of which all the dissimilarity values are taken into consideration.

In general, the HaDiPCS procedure as referred in figure 4.3 while it was effective, it was time consuming when the threshold reached above the value of 0.6. The computational cost and time consumption of implementation of the method is closely related to the amount of pairs of filters that have HD matrix values below the threshold. It is natural that when the number of filters is larger, the time the select filters for pruning will augment because of the 3-pair pruning method, especially in deeper layers where the values of HD dissimilarities tend to be quite low and the number of filters and channels is higher. In general, the average accuracy of all the pruned models is 0.695 and it is higher than the accuracy of the baseline model which is 0.6787 concluding that while some of the pruned models have lower accuracy, the HaDiPCS method increases the model's accuracy on the whole.

5.3 MNIST Evaluation Results

After the HaDiPCS method is implemented using a percentile-based threshold, it is clear that accuracy and loss remained the same with 1 exception, the threshold value of 70, representing the 70th percentile. Even in this specific value, the accuracy of the model does not drop below 98% of the accuracy while the loss of the pruned model is also impressively low.

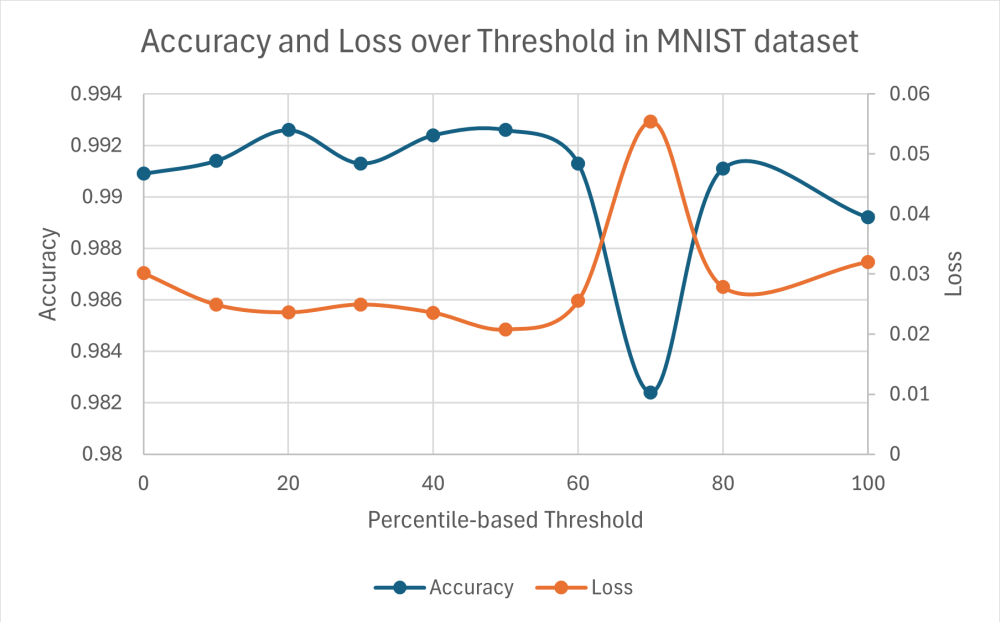


Figure 5.6: Accuracy and Loss that occurred after pruning the model with different percentile-based thresholds in MNIST dataset.

The inference time of the pruned model drops significantly reaching half of the baseline model's inference time from the early threshold of the 70th percentile while the method in the CIFAR-10 dataset reached a relevant decrease only in the maximum threshold value. This is a remarkable result as it shows that the technique can lower the inference time even in earlier threshold values which means that the pruning will not require lots of computations to conclude with a desirably faster pruned model.

Furthermore, the FLOPs and Paremeters of the pruned models, showcase a similar behavior with the reduction of both, as the threshold increases. The FLOPs in the pruned model when applying the threshold at the 100th percentile, present a tremendous drop, reducing the model's operations to far below one fifth of the baseline model's FLOPs (very close to one sixth of the original FLOPs of the baseline model).

The table 5.1 shows the performance of HaDiPCS in various datasets comparing the ac-

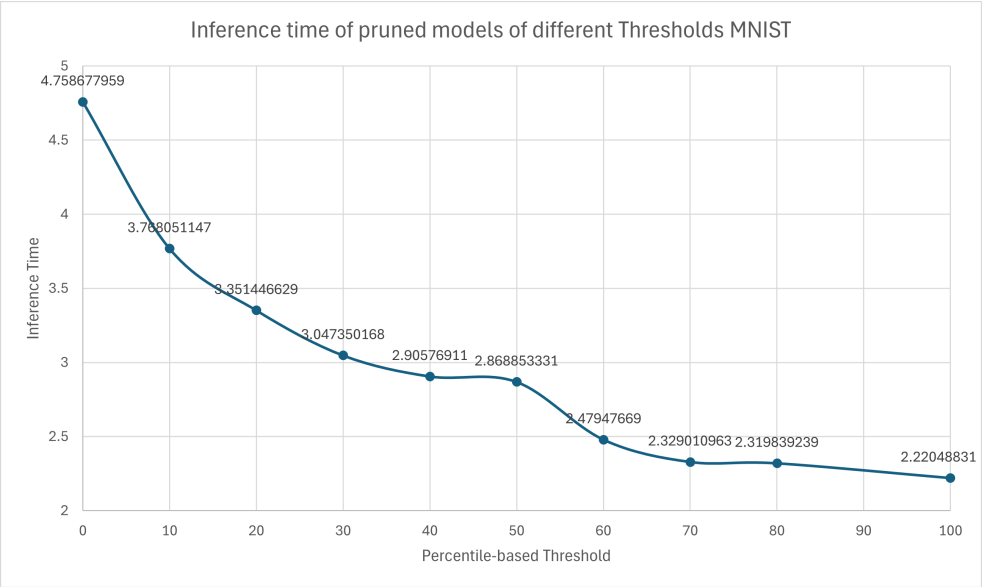


Figure 5.7: Inference time of models that resulted from pruning the initial baseline model with different percentile-based threshold values in MNIST dataset

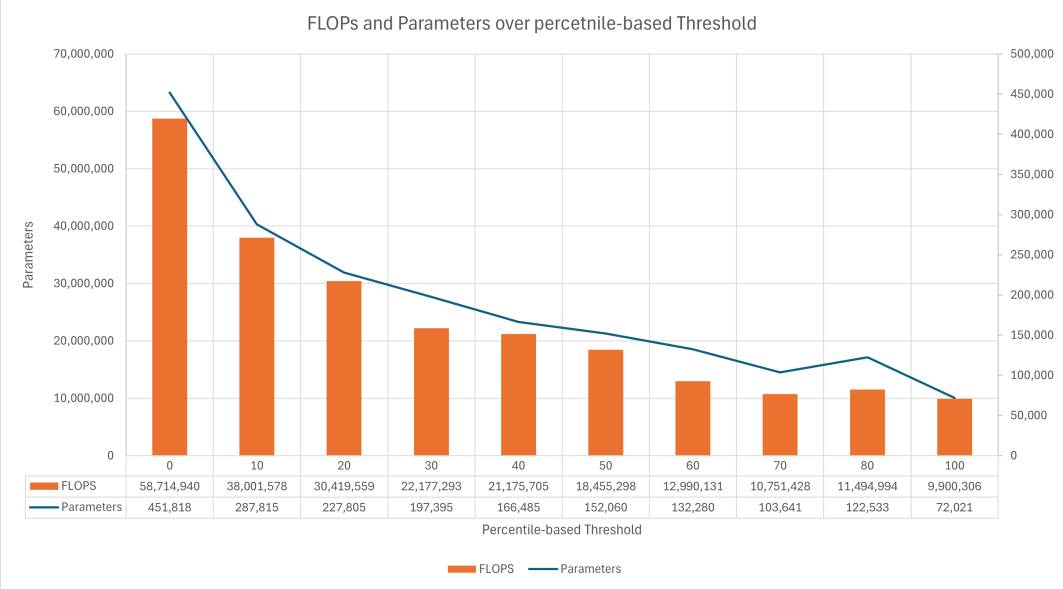


Figure 5.8: FLOPs and Parameters of pruned models for various percentile-based thresholds in MNIST dataset

curacy of the baseline unpruned model to the accuracy of a pruned model with the threshold set at only the 40th percentile of the dissimilarity values. In the majority of the datasets, the pruned model has a higher accuracy than the baseline model and in the only case the accuracy drops is in the SVHN dataset in which the accuracy is reduced by less than 0,14%, a truly minimal decrease. The percentage of trainable parameters that have been reduced in the pruned model is relatively high compared to the low threshold, proving that HaDiPCS works

well even without the need to take all filter dissimilarity values into consideration and consume a large amount of time accompanied by a high threshold. The trainable parameters of the model present a significant decrease with the maximum observed in the MNIST-C dataset in the 'brightness' noise type of the input images. In this case the model's size is decreased by as high as 66.37% maintaining an increased accuracy. The large decrease in the trainable parameters of the model is of high importance as less parameters indicate lower memory footprint, lower inference time and less FLOPs.

Table 5.1: Evaluation in various Datasets

Dataset	**Baseline Model Accuracy (%)**	**Pruned Model Accuracy (%)**	**Pruned Model Parameter Decrease (%)**
CIFAR-10	67.87	69.94 ↑	48.64
MNIST	99.09	99.24 ↑	63.15
Fashion-MNIST	90	90.40 ↑	60.6
SVHN	89.91	89.77 ↓	63.64
MNIST-C 'brightness'	98.85	98.87 ↑	66.37
MNIST-C 'fog'	98.8	99.14 ↑	65.13
MNIST-C 'shot_noise'	98.41	98.65 ↑	63.82
MNIST-C 'canny_edges'	98.44	98.60 ↑	65.22

Chapter 6

Conclusions and future directions

6.1 Conclusions

Convolutional Neural Networks as they become deeper, can present several challenges embedded in their computational intensity and complexity. They may consist of hundreds of thousands or even millions of trainable parameters requiring substantial computational resources for the training of these parameters and the intensive operations performed in the convolutional layers. This not only makes training time-consuming but also imposes significant costs associated with hardware infrastructure and electricity consumption.

HaDiPCS pruning technique attempts to prune filters based on their dissimilarity using the Hausdorff Distance. After the Hausdorff Distance between all channels of 2 filters is computed (in pairs of 2 filters), it is inserted into a matrix that contains the dissimilarity values between every filter combination inside each layer. Then all filter pairs that have dissimilarity values below a specific custom threshold, are examined by the 3-pair and 2-pair pruning methods. These methods conclude with a list of filters that are redundant since they seem to learn very common features to other filters chosen to remain in the layer. This happens in order for the image attribute commonly learned by several filters, to be represented by at least one remaining filter inside the layer. Then this list is used to prune the filters in the specific convolutional layer of the model. This procedure is repeated for each convolutional layer of the model and then the model is retrained for a few epochs in the dataset.

In conclusion, the HaDiPCS method has displayed remarkable efficiency, combining a high accuracy of the pruned model which sometimes surpasses the original model's accuracy, and a significant reduction of the initial model's parameters. Moreover, when the thresh-

old is set at a relatively low value with the optimal one being slightly below the threshold value of 0.5 for range-based thresholds and the median value for percentile-based thresholds, HaDiPCS achieves a pre-eminent balance between the algorithm's runtime and inference time reduction of the pruned model while it also decreases considerably the Floating Point Operations of the model. The strength of the HaDiPCS method lies in each ability to check which filters are redundant inside each specific layer while also taking into consideration that filters probably needed to preserve a high accuracy should not be removed. Thus, it combines the high accuracy of the pruned models which is usually even higher then the baseline model along with a considerable drop in the model's trainable parameters which is strongly associated with a lower memory footprint and a lower inference time. Very few, if any, pruning techniques manage to achieve this combination.

The most astounding results were noticed when the threshold was set at the highest possible values. On that occasion, the technique considered every filter of each layer for pruning no matter how important they proved to be for the model's performance. As it is to be expected, the FLOPs and Inference time of the pruned model in the highest threshold values presented a tremendous drop resulting in a pruned model that consumes half the inference time and executes 1/6 of the FLOPs of the original model. Despite the astonishing results presented above, the cost of the algorithm's runtime remains a serious problem as it is a critical consideration for its implementation in real-time applications. It requires at least 30-35 hours on CPU based machines to execute the examination of all filters inside each layer for pruning, with deeper layers that contain numerous filters requiring the most time. Thus, the threshold value must be carefully selected, as it can greatly impact the algorithm's execution time, particularly when a GPU is not available.

6.2 Future Directions

For now, the algorithm is in its early form and although its effectiveness is established, it is not in its optimal state. The next step will be to experiment with several optimization techniques with the goal of reducing the algorithm's computational cost and runtime. Some indicative techniques are the replacement of lists and tuples used in the 3-pair and 2-pair pruning techniques with NumPy arrays to save memory and speed up calculations, moreover the HaDiPCS could employ vectorization over for loops as this specific technique has proved

to be faster and more efficient as it involves the utilization of pre-compiled operations on NumPy arrays and although the runtime difference is usually minor, in this specific procedure it can improve the execution time considerably [52].

Another alternative implementation of the HaDiPCS method is the creation of a Hausdorff Distance Matrix that contains only the maximum Hausdorff Distance as displayed in equation 4 (&). In this case the matrix will be symmetrical, and each pair of filters will only be examined once instead of creating a matrix that contains different dissimilarity values for each pair of filters and examining if both of them have dissimilarity values below the threshold. This technique will be ideal for low-resource devices, as its computational cost will be tremendously decreased. As explained in 4, several pairs of filters have very different dissimilarity values which means that the choice of the maximum out of their values will deprive the technique of significant information about the importance of specific filters. This piece of information will be utilized in a later experimentation using the HaDiPCS technique in which the pairs of filters with large divergence between their dissimilarity values will be pruned to determine if the presence of them both actually offers insight into the model. Moreover, an entirely different strategy could be implemented in which the pairs of filters with large deviation of dissimilarity values will be compared to determine the specific filter that causes the problem and prune it. This way, a larger amount of filters will be pruned and perhaps the technique will result in a shorter and stronger model.

After the experimentation with the former optimizations, the HaDiPCS technique will be implemented in deeper CNNs such as VGG-16 [53] using larger datasets with more classes such as CIFAR-100 [50] and ImageNet [54]. Moreover, models like ResNet [55] and DenseNet [56] will be implemented all with the usage of a GPU for the comparison against latest and more robust state-of-the-art pruning techniques.

Bibliography

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[4] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[5] Pi, your personal ai. `https://pi.ai/discover`. Ημερομηνία πρόσβασης: 20-06-2024.

[6] Kay Lehnert. Ai insights into theoretical physics and the swampland program: A journey through the cosmos with chatgpt, 2023.

[7] Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, Zihao Wu, Lin Zhao, Dajiang Zhu, Xiang Li, Ning Qiang, Dingang Shen, Tianming Liu, and Bao Ge. Summary of chatgpt-related research and perspective towards the future of large language models. *Meta-Radiology*, 1(2):100017, September 2023.

[8] Chenyang Lyu, Jitao Xu, and Longyue Wang. New trends in machine translation using large language models: Case examples with chatgpt. *arXiv preprint arXiv:2305.01181*, 2023.

[9] Giriprasad Sridhara, Sourav Mazumdar, et al. Chatgpt: A study on its utility for ubiquitous software engineering tasks. *arXiv preprint arXiv:2305.16837*, 2023.

[10] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 2023.

[11] Prabin Sharma, Kisan Thapa, Prastab Dhakal, Mala Deep Upadhaya, Santosh Adhikari, and Salik Ram Khanal. Performance of chatgpt on usmle: Unlocking the potential of large language models for ai-assisted medical education. *arXiv preprint arXiv:2307.00112*, 2023.

[12] Yue Zhang, Jiong Wu, Yilong Liu, Yifan Chen, Wei Chen, Ed X Wu, Chunming Li, and Xiaoying Tang. A deep learning framework for pancreas segmentation with multi-atlas registration and 3d level-set. *Medical Image Analysis*, 68:101884, 2021.

[13] Titus J Brinker, Achim Hekler, Alexander H Enk, Joachim Klode, Axel Hauschild, Carola Berking, Bastian Schilling, Sebastian Haferkamp, Dirk Schadendorf, Tim Holland-Letz, et al. Deep learning outperformed 136 of 157 dermatologists in a head-to-head dermoscopic melanoma image classification task. *European Journal of Cancer*, 113:47–54, 2019.

[14] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[15] Rodolfo M Pereira, Diego Bertolini, Lucas O Teixeira, Carlos N Silla Jr, and Yandre MG Costa. Covid-19 identification in chest x-ray images on flat and hierarchical classification scenarios. *Computer methods and programs in biomedicine*, 194:105532, 2020.

[16] Pedro Celard, Eva Lorenzo Iglesias, José Manuel Sorribes-Fdez, Rubén Romero, A Seara Vieira, and Lourdes Borrajo. A survey on deep learning applied to medical

images: from simple artificial neural networks to generative models. *Neural Computing and Applications*, 35(3):2291–2323, 2023.

[17] Run Su, Deyun Zhang, Jinhuai Liu, and Chuandong Cheng. Msu-net: Multi-scale u-net for 2d medical image segmentation. *Frontiers in Genetics*, 12:639930, 2021.

[18] Fabian Isensee, Paul F Jaeger, Simon AA Kohl, Jens Petersen, and Klaus H Maier-Hein. nnu-net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods*, 18(2):203–211, 2021.

[19] Xi Ouyang, Jiayu Huo, Liming Xia, Fei Shan, Jun Liu, Zhanhao Mo, Fuhua Yan, Zhongxiang Ding, Qi Yang, Bin Song, et al. Dual-sampling attention network for diagnosis of covid-19 from community acquired pneumonia. *IEEE Transactions on Medical Imaging*, 39(8):2595–2605, 2020.

[20] Won-Joon Do, Sunghun Seo, Yoseob Han, Jong Chul Ye, Seung Hong Choi, and Sung-Hong Park. Reconstruction of multicontrast mr images through deep learning. *Medical physics*, 47(3):983–997, 2020.

[21] Andrii Pozaruk, Kamlesh Pawar, Shenpeng Li, Alexandra Carey, Jeremy Cheng, Viswanath P Sudarshan, Marian Cholewa, Jeremy Grummet, Zhaolin Chen, and Gary Egan. Augmented deep learning model for improved quantitative accuracy of mr-based pet attenuation correction in psma pet-mri prostate imaging. *European journal of nuclear medicine and molecular imaging*, 48:9–20, 2021.

[22] Navid Ghassemi, Afshin Shoeibi, and Modjtaba Rouhani. Deep neural network with generative adversarial networks pre-training for brain tumor classification based on mr images. *Biomedical Signal Processing and Control*, 57:101678, 2020.

[23] Zhiwei Qin, Zhao Liu, Ping Zhu, and Yongbo Xue. A gan-based image synthesis method for skin lesion classification. *Computer Methods and Programs in Biomedicine*, 195:105568, 2020.

[24] Xiao Zhou, Shangran Qiu, Prajakta S Joshi, Chonghua Xue, Ronald J Killiany, Asim Z Mian, Sang P Chin, Rhoda Au, and Vijaya B Kolachalama. Enhancing magnetic resonance imaging-driven alzheimer's disease classification performance using generative adversarial learning. *Alzheimer's research & therapy*, 13(1):1–11, 2021.

[25] Takahiro Nakao, Shouhei Hanaoka, Yukihiro Nomura, Masaki Murata, Tomomi Takenaga, Soichiro Miki, Takeyuki Watadani, Takeharu Yoshikawa, Naoto Hayashi, and Osamu Abe. Unsupervised deep anomaly detection in chest radiographs. *Journal of Digital Imaging*, 34:418–427, 2021.

[26] Sergio Naval Marimont and Giacomo Tarroni. Anomaly detection through latent space restoration using vector quantized variational autoencoders. In *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pages 1764–1767. IEEE, 2021.

[27] Jan Egger, Christina Gsaxner, Antonio Pepe, Kelsey L Pomykala, Frederic Jonske, Manuel Kurz, Jianning Li, and Jens Kleesiek. Medical deep learning—a systematic meta-review. *Computer methods and programs in biomedicine*, 221:106874, 2022.

[28] Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A Roberts. The unreasonable ineffectiveness of the deeper layers. *arXiv preprint arXiv:2403.17887*, 2024.

[29] Hongrong Cheng, Miao Zhang, and Javen Qinfeng Shi. A survey on deep neural network pruning-taxonomy, comparison, analysis, and recommendations. *arXiv preprint arXiv:2308.06767*, 2023.

[30] Yang He and Lingao Xiao. Structured pruning for deep convolutional neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[31] Paul Wimmer, Jens Mehnert, and Alexandru Paul Condurache. Dimensionality reduced training by pruning and freezing parts of a deep neural network: a survey. *Artificial Intelligence Review*, 56(12):14257–14295, 2023.

[32] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4340–4349, 2019.

[33] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1389–1397, 2017.

[34] Xiaohan Ding, Guiguang Ding, Yuchen Guo, Jungong Han, and Chenggang Yan. Approximated oracle filter pruning for destructive cnn width optimization. In *International Conference on Machine Learning*, pages 1607–1616. PMLR, 2019.

[35] Ruichi Yu, Ang Li, Chun-Fu Chen, Jui-Hsin Lai, Vlad I Morariu, Xintong Han, Mingfei Gao, Ching-Yung Lin, and Larry S Davis. Nisp: Pruning networks using neuron importance score propagation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9194–9203, 2018.

[36] Xiaohan Ding, Tianxiang Hao, Jianchao Tan, Ji Liu, Jungong Han, Yuchen Guo, and Guiguang Ding. Resrep: Lossless cnn pruning via decoupling remembering and forgetting. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4510–4520, 2021.

[37] Huan Wang, Can Qin, Yulun Zhang, and Yun Fu. Neural pruning via growing regularization. *arXiv preprint arXiv:2012.09243*, 2020.

[38] William J Vetter. Matrix calculus operations and taylor expansions. *SIAM review*, 15(2):352–369, 1973.

[39] Stephen E Fienberg. When did bayesian inference become" bayesian"? 2006.

[40] Zih-Syuan Huang and Ching-pei Lee. Training structured neural networks through manifold identification and variance reduction. *arXiv preprint arXiv:2112.02612*, 2021.

[41] Andreas Chouliaras, Evangelia Fragkou, and Dimitrios Katsaros. Feed forward neural network sparsificationwith dynamic pruning. In *Proceedings of the 25th pan-hellenic conference on informatics*, pages 12–17, 2021.

[42] Evangelia Fragkou, Marianna Koultouki, and Dimitrios Katsaros. Model reduction of feed forward neural networks for resource-constrained devices. *Applied Intelligence*, 53(11):14102–14127, 2023.

[43] Abdel Aziz Taha and Allan Hanbury. An efficient algorithm for calculating the exact hausdorff distance. *IEEE transactions on pattern analysis and machine intelligence*, 37(11):2153–2163, 2015.

[44] Daniel P Huttenlocher, Gregory A. Klanderman, and William J Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on pattern analysis and machine intelligence*, 15(9):850–863, 1993.

[45] Eric Billet, Andriy Fedorov, and Nikos Chrisochoides. The use of robust local hausdorff distances in accuracy assessment for image alignment of brain mri. *Insight Journal (January-June 2008), http://hdl. handle. net/1926/1354*, 2008.

[46] Sarana Nutanong, Edwin H Jacox, and Hanan Samet. An incremental hausdorff distance calculation algorithm. *Proceedings of the VLDB Endowment*, 4(8):506–517, 2011.

[47] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 7. Granada, Spain, 2011.

[48] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[49] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142, 2012.

[50] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[51] Norman Mu and Justin Gilmer. Mnist-c: A robustness benchmark for computer vision. *arXiv preprint arXiv:1906.02337*, 2019.

[52] Vectorization in python- an alternative to python loops. `https://medium.com/pythoneers/vectorization-in-python-an-alternative-to-python-loops-2728d6d7cd3e`. Ημερομηνία πρόσβασης: 20-06-2024.

[53] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[54] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[55] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[56] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks, 2018.