G. E. DAĞI

A STUDY IN THE IMPLEMENTATION OF CONVOLUTIONAL NEURAL NETWORK FOR IMAGE CLASSIFICATION IN FREQUENCY DOMAIN

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

GÖKTUĞ ERDEM DAĞI

A MASTER OF SCIENCE THESIS
IN
THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

JUNE 2024

ATILIM UNIVERSITY 2024

A STUDY IN THE IMPLEMENTATION OF CONVOLUTIONAL NEURAL NETWORK FOR IMAGE CLASSIFICATION IN FREQUENCY DOMAIN

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

BY

GÖKTUĞ ERDEM DAĞI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

JUNE 2024

Approval of the Graduate School of Natural and Applied Sciences, Atılım University.

Prof. Dr. Ender KESKİNKILIÇ

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of **Master of Science in Electrical and Electronics Engineering Department, Atılım University**.

Prof. Dr. Reşat Özgür DORUK

Head of Department

This is to certify that we have read the thesis A STUDY IN THE IMPLEMENTATION OF CONVOLUTIONAL NEURAL NETWORK FOR IMAGE CLASSIFICATION IN FREQUENCY DOMAIN submitted by GÖKTUĞ ERDEM DAĞI and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Dr. Öğr. Üyesi Hakan TORA
Co-Supervisor

Dr. Öğr. Üyesi Erhan GÖKÇAY
Supervisor

Examining Committee Members:

Dr. Öğr. Üyesi İbrahim Baran USLU
Electrical and Electronics Engineering
Ostim Technical University

Dr. Öğr. Üyesi Erhan GÖKÇAY
Software Engineering
Atılım University

Assoc. Prof. Dr. Yaser DALVEREN
Electrical and Electronics Engineering
Atılım University

Date: June 24, 2024

I declare and guarantee that all data, knowledge and information in this document has been obtained, processed and presented in accordance with academic rules and ethical conduct. Based on these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Göktuğ Erdem DAĞI

Signature :

ABSTRACT

A STUDY IN THE IMPLEMENTATION OF CONVOLUTIONAL NEURAL NETWORK FOR IMAGE CLASSIFICATION IN FREQUENCY DOMAIN

Göktuğ Erdem DAĞI

M.S., Department of Electrical and Electronics Engineering

Supervisor : Dr. Öğr. Üyesi Erhan GÖKÇAY

July 2024, 44 pages

In recent years, Convolutional Neural Networks (CNNs) have achieved remarkable success in various image processing and computer vision tasks. Traditional CNNs operate directly on spatial domain images. However, the frequency domain representation of images obtained through Fast Fourier Transform (FFT) offers unique advantages, such as decorrelation of pixel values and potential reduction in computational complexity. This thesis aims to investigate the effects of using FFT-transformed images as input to CNN algorithms to enhance image classification and recognition accuracy. The research begins with a comprehensive examination of the theoretical foundations and properties of FFT. It then explores the integration of FFT in preprocessing pipelines for CNNs. By converting input images from the spatial domain to the frequency domain, we hypothesize that CNNs can learn more efficiently by focusing on the most significant frequency components, thereby potentially improving convergence rates and overall performance. Experiments were con- ducted using various benchmark datasets, including CIFAR-10(Canadian Institute For Advanced Research), MNIST(Modified National Institute of Standards and

Technology)-Digits, and MNIST-Fashion, to evaluate the efficacy of this approach. FFT-transformed images were fed into various CNN architectures, and the results were compared with those obtained using traditional spatial domain inputs. Metrics such as classification accuracy, training time, and computational resource utilization were meticulously analyzed. The results indicate that FFT-based preprocessing can lead to improvements in classification accuracy, particularly in scenarios where the datasets contain high-frequency noise or redundant information. However, the benefits varied across different datasets and network architectures, suggesting that the effectiveness of FFT preprocessing may be context dependent. In conclusion, this thesis demonstrates that incorporating FFT preprocessing into CNN work- flows holds promise for enhancing image processing tasks. The findings suggest avenues for future research, including the development of hybrid models that leverage both spatial and frequency domain information and the application of FFT-based techniques to other types of neural networks and machine learning algorithms. This study contributes to a broader understanding of how frequency domain analysis can be synergistically integrated with deep learning methodologies to advance the field of computer vision.

Keywords: Machine Learning, Image Classification, Frequency Domain, Deep Learning.

ÖZ

FREKANS ALANINDA GÖRÜNTÜ SINIFLANDIRMA İÇİN KONVOLÜSYONEL SİNİR AĞLARININ UYGULANMASI

Dağı, Göktuğ Erdem
Yüksek Lisans, Elektrik Electronik Mühendisliği Bölümü
Tez Yöneticisi : Dr. Öğr. Erhan Gökçay

Haziran 2024, 44 sayfa

Bu tezde, Evrişimsel Sinir Ağları (CNN'ler) son yıllarda çeşitli görüntü işleme ve bilgisayarlı görme görevlerinde dikkate değer başarılar elde etmiştir. Geleneksel CNN'ler doğrudan uzaysal alan görüntüleri üzerinde çalışır. Bununla birlikte, Hızlı Fourier Dönüşümü (FFT) yoluyla elde edilen görüntülerin frekans alanı gösterimi, piksel değerlerinin ilişkisizleştirilmesi ve hesaplama karmaşıklığında potansiyel azalma gibi benzersiz avantajlar sunar. Bu tez, görüntü sınıflandırmasını ve tanıma doğruluğunu artırmak için FFT ile dönüştürülmüş görüntülerin CNN algoritmalarına girdi olarak kullanılmasının etkilerini araştırmayı amaçlamaktadır. Araştırma, FFT'nin teorik temellerinin ve özelliklerinin kapsamlı bir incelemesiyle başlıyor. Daha sonra CNN'ler için ön işleme ardışık düzenlerinde FFT'nin entegrasyonunu araştırıyor. Giriş görüntülerini uzamsal alandan frekans alanına dönüştürerek, CNN'lerin en önemli frekans bileşenlerine odaklanarak daha verimli öğrenebileceğini, dolayısıyla yakınsama oranlarını ve genel performansı potansiyel olarak iyileştirebileceğini varsayıyoruz. Bunun etkinliğini değerlendirmek için CIFAR-10 (Kanada İleri Araştırma Enstitüsü), MNIST (Modifiye Ulusal Standartlar ve Teknoloji Enstitüsü)-Digits ve MNIST-Fashion dahil olmak üzere çeşitli kıyaslama veri setleri kullanılarak deneyler gerçekleştirildi. yaklaşmak. FFT ile dönüştürülmüş görüntüler çeşitli CNN mimarilerine beslendi ve sonuçlar, geleneksel uzaysal alan girdileri kullanılarak elde

edilenlerle karşılaştırıldı. Sınıflandırma doğruluğu, eğitim süresi ve hesaplamalı kaynak kullanımı gibi ölçümler titizlikle analiz edildi. Sonuçlar, FFT tabanlı ön işlemenin, özellikle veri kümelerinin yüksek frekanslı gürültü veya gereksiz bilgi içerdiği senaryolarda, sınıflandırma doğruluğunda iyileştirmelere yol açabileceğini göstermektedir. Ancak faydaların farklı veri kümeleri ve ağ mimarileri arasında farklılık göstermesi, FFT ön işlemenin etkililiğinin bağlama bağlı olabileceğini düşündürmektedir. Sonuç olarak bu tez, FFT ön işlemesinin CNN iş akışlarına dahil edilmesinin görüntü işleme görevlerini geliştirme konusunda umut vaat ettiğini göstermektedir. Bulgular, hem uzaysal hem de frekans alanı bilgisinden yararlanan hibrit modellerin geliştirilmesi ve FFT tabanlı tekniklerin diğer sinir ağı türlerine ve makine öğrenimi algoritmalarına uygulanması da dahil olmak üzere gelecekteki araştırmalar için yollar önermektedir. Bu çalışma, bilgisayarlı görme alanını geliştirmek için frekans alanı analizinin derin öğrenme metodolojileriyle nasıl sinerjik olarak entegre edilebileceğinin daha geniş bir şekilde anlaşılmasına katkıda bulunmaktadır.

Anahtar Kelimeler: Makine Öğrenmesi, Görüntü Sınıflandırması, Frekans Alanı Derin Öğrenme.

To my family...

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to Dr. Öğr. Üyesi Erhan Gökçay and Dr. Öğr. Üyesi Hakan Tora for his priceless and in-depth guidance throughout the thesis period. Working with him was a great honour and opportunity. Over every step of my academic research, I have been inspired by his immense knowledge and wealth of experience.

I am deeply indebted to Dr. Öğr. Üyesi İbrahim Baran Uslu for his valuable comments, concern, and patience. Regular discussions and meetings have served as the foundation for the completeness of this thesis.

I would like to extend my sincere thanks to Dr. Öğr. Üyesi Erhan Gökçay and Dr. Öğr. Üyesi Hakan Tora for participating on my thesis committee and offering insightful feedback and recommendations.

Finally, I would like to thank my family and friends for their unwavering support and understanding throughout this journey.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

*Adagrad :*Adaptive Gradient Algorithm

Adam : Adaptive Moment Estimation

AI : Artificial Intelligence

CIFAR : Canadian Institute For Advanced Research

CNN : Convolutional Neural Networks

DFT : Discrete Fourier Transform

ELU : Exponential Linear Unit
FTT : Fast Fourier Transform

k – NN : k-Nearest Neighbor

MNIST : Modified National Institute of Standards and Technology

MSE : Mean Squared Error

NAS : Neural Architecture Search

RGB : Red, Blue, Green

ReLU : Rectifier Liner Unit
SGD : Stochastic Gradient Descent

SIFT : Scale-Invariant Feature Transform

SURF : Speeded-Up Robust Features

SVMs : Support Vector Machines

Tanh : Hyperbolic Tangent

CHAPTER 1

INTRODUCTION

Artificial intelligence (AI) has become a field that has developed rapidly in recent years and has made a huge impact in many industrial and academic fields. Artificial Intelligence (AI) refers to the simulation of human intelligence and behavior in machines programmed to think and imitate human actions. In particular, advances in image processing and analysis have demonstrated the power and potential of AI. Extracting meaningful information from image data has found applications in medicine, security, automotive industry, retail, and many other fields. These machines are designed to perform tasks that often require human intelligence, such as visual perception, speech recognition, decision-making, and language translation. In this context, the use of artificial intelligence techniques, especially deep learning models such as Convolutional Neural Networks (CNN), has brought revolutionary progress in the field of image processing. Artificial intelligence systems can analyze large amounts of data, recognize patterns, and make predictions or decisions based on this data.

Today, artificial intelligence (AI) is one of the fastest growing and most exciting fields of computer science. Artificial intelligence aims to simulate human like intelligence and behavior using computers and attempts to achieve this goal through a set of algorithms and techniques. One of the most important of these techniques is artificial neural networks, which include a subfield known as deep learning.

Deep learning is achieved through the use of complex artificial neural networks trained on large amounts of data. These networks simulate the process of learning from data and discovering complex patterns, similar to how the human brain works. One of the most effective and widely used types of deep learning models is known as

Convolutional Neural Networks (CNNs). CNNs are a type of artificial neural networks that demonstrate superior performance, especially when working with image data.

Image processing can be defined as the field that enables the computer to inter- pret, understand and process an image. Image processing techniques are widely used in various fields such as object recognition, facial recognition, medical imaging, autonomous vehicles and security systems. When combined with artificial intelligence techniques, image processing becomes more complex and effective. In this context, deep learning models such as CNNs show superior performance in various tasks in the field of image processing. These advances in the field of image processing have especially allowed to overcome the limitations of traditional pixel-based methods and develop new approaches to address more complex tasks.

Akwaboah's study involved developing and training three different CNN models in the spatial domain to classify CIFAR-10 images. These models varied in their convolutional filter sizes, pooling layers, and the use of dropout regularization[1].

Model 1: Achieved a test accuracy of 72.81%.

Model 2: Achieved a test accuracy of 67.07%.

Model 3: Achieved a test accuracy of 75.43% .

Adeyinka's research focused on optimizing CIFAR-10 classification by testing various CNN models with different depths and configurations.

Advanced architectures like ResNet and DenseNet were utilized, achieving test accuracies exceeding 90%. These models leverage deeper layers and skip connections, significantly enhancing model generalization and robustness[2].

Hengyue Pan proposed a novel approach by training a CNN model, named CEM- Net, in the frequency domain. The approach simplifies the convolution operation, making it easier to parallelize by replacing it with element-wise multiplication[3].

CEMNet introduced several enhancements, including a weight fixation mechanism to mitigate overfitting, and adaptations of batch normalization, Leaky ReLU, and dropout layers for the frequency domain. Experimental results demonstrated that CEMNet could achieve over 70% accuracy on the CIFAR- 10 dataset[3].

Classification was performed using two types of images: RGB images and Fourier-transformed images. Sophie T¨otterstrom achieved an accuracy of 79.23% for RGB images and 38.78% for frequency domain images on the CIFAR-10 dataset [4]. In this study, the fields of artificial intelligence and image processing are discussed and the importance of combining these fields to develop integrated methods is emphasized. It aims to examine how artificial intelligence techniques can be used, especially in the field of image processing, and how more powerful and efficient results can be obtained by combining these techniques. In particular, it will focus on training frequency representations obtained by processing input images with Fourier Transform (FFT) on CNN models. FFT is a mathematical procedure used to re- veal the frequency components of an image. This process enables the representation of different frequency components of images and offers a different perspective from traditional pixel-based image processing methods.

The main purpose of this thesis is to investigate how the frequency components obtained by taking the FFT of the input images can be integrated into the learning process of CNN models and how this integration can affect the image classification performance. Compared to traditional pixel-based image processing methods, the advantages of FFT representations to CNN models and the general usability of the results will be emphasized.

CHAPTER 2

BACKGROUND

Since the images will be represented in Fourier Transform, information about FourierTransform will be given in the following sections.

2.1.Fourier Transform

The Fourier Transform is a mathematical tool used in various fields, especially in signal processing, image analysis, and physics. It essentially breaks down a function (often a waveform or a signal) into its constituent frequencies. This transformation enables the representation of complex signals in terms of simpler sinusoidal functions[5].

The Fourier Transform takes a function in the time (or spatial) domain and con- verts it into a function in the frequency domain. In simpler terms, it decomposes a signal into its constituent frequencies and their respective amplitudes and phases.

The mathematical formula for the Fourier Transform of a function f(t) is given by:

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{j\omega t} \partial t$$

F (*ω*)**:** This represents the Fourier Transform of the function *f* (*t*). It gives us the amplitude and phase of the sinusoidal components of the function *f* (*t*) at different frequencies *ω*. *ω* is the angular frequency in radians per unit time.

f (*t*)**:** This represents the original function in the time domain. It's the function that you're transforming from the time domain to the frequency domain.

*t***:** This is the variable representing time. In the context of the Fourier Trans- form, *t*

ranges from negative infinity to positive infinity, covering the entire time domain.

$e^{-j\omega t}$**:** This term is the complex exponential function. It represents sinusoidal oscillations at frequency ω and phase $\omega * t$.

ω**:** This represents the angular frequency in radians per unit time. It's the frequency variable in the frequency domain. The Fourier Transform gives us information about how much of each frequency component ω is present in the original signal.

The basic idea behind the Fourier Transform is that any periodic function can be represented as a sum of sinusoidal functions with different frequencies. This decomposition allows us to analyze and manipulate signals more effectively, facilitating tasks such as filtering, compression, and pattern recognition.

There are different versions of the Fourier Transform, such as the Discrete Fourier Transform (DFT) and the Fast Fourier Transform (FFT), which are computationally efficient algorithms used to calculate the Fourier Transform of discrete signals or data points. These algorithms have widespread applications in digital signal processing and are used extensively in fields like telecommunications, audio processing, and image processing. Fast Fourier Transform is used in this thesis.

2.1.1.Discrete Fourier Transform (DFT)

Discrete Fourier Transform (DFT) is a mathematical transformation used to analyze the frequencies present in a discrete set of data points. It is particularly useful for analyzing signals sampled at discrete time intervals, such as digital signals in audio processing, image processing, and telecommunications[5].

Discrete Fourier Transform (DFT) is a mathematical operation used to decompose a signal (or a series of data points) into its frequency components. It takes a finite sequence of equally spaced data points and transforms it from the time (or spatial) domain to the frequency domain. In other words, it converts a numerical sequence representing a signal into another numerical sequence representing the frequency content of the signal.

The mathematical formula for the Discrete Fourier Transform of a sequence $x[n]$ of length N is given by:

Here's what the variables represent:

$x[n]$: The input in the time domain. It is the n-th sample of the original time signal.

$X[k]$: The output in the frequency domain. It is the k-th frequency component of the DFT.

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N}$$

N : The total number of samples in the signal.

n: The time index, which takes integer values from 0 to $\boldsymbol{N-1}$.

k: The frequency index, which takes integer values from 0 to $\boldsymbol{N-1}$.

Addition calculates the contribution of each frequency component k to the original $x[n]$ sequence. The result is a sequence of complex numbers $X[k]$ that provides information about the amplitudes and phases of the sinusoidal components of $x[n]$ at different frequencies.

Mathematically, the DFT of an N -element signal is expressed as an N -element array, which contains the frequency components of the signal. DFT shows the presence and intensity of specific frequency components of the signal. This is crucial for analyzing and processing the frequency components of a signal.

DFT is used in various fields such as digital signal processing, data compression, spectral analysis, and many other applications. Particularly, the FFT (Fast Fourier Transform), which is a fast algorithm for computing DFT, enables efficient computation of DFT and is widely used in many real-time applications.

2.1.2.Fast Fourier Transform (FFT)

The Fast Fourier Transform (FFT) is an algorithm used to efficiently compute the Discrete Fourier Transform (DFT) and its inverse. It significantly reduces the computational complexity of calculating the DFT, especially for large data sets, making it feasible to perform Fourier analysis in real-time and on digital computers.

The FFT algorithm was developed by Cooley and Tukey in 1965 and has since become

one of the most important numerical algorithms, widely used in various fields such as signal processing, image processing, audio processing, and many others[6].

The basic idea behind the FFT is to exploit the symmetry and periodicity properties of the discrete Fourier transform to divide the computation into smaller, more manageable subproblems. By recursively decomposing the DFT into smaller DFTs, the FFT algorithm reduces the number of arithmetic operations required from $O(N^2)$ to $O(N \log N)$, where N is the number of data points in the sequence.

The Fast Fourier Transform (FFT) is an algorithm for efficiently computing the Discrete Fourier Transform (DFT) and its inverse. The mathematical formula for the DFT was provided in the previous section. Below we will explain how the FFT algorithm works and what its variables represent.

$x[n]$**:** the input sequence of length N.

$X[k]$**:** the output sequence of the DFT.

N **:** the number of data points in the sequence.

The FFT algorithm can be described by the following recursive formula:

$$X[k] = X_{\text{even}}[k] + e^{-j2\pi k/N} \cdot X_{\text{odd}}[k]$$

Where:

$X_{\textbf{even}}[k]$**:** the DFT of the even-indexed elements of $x[n]$.

$X_{\textbf{odd}}[k]$**:** the DFT of the odd-indexed elements of $x[n]$.

$e^{-j2\pi k/N}$ **:** the twiddle factor, which represents the phase shift introduced by the frequency k in the DFT calculation.

The FFT algorithm divides the DFT computation recursively into smaller sub-problems until it reaches the base case of a single-point DFT, which is trivial to compute. Then, it combines the results of the smaller DFTs to obtain the final DFT of the entire sequence.

2.2.Previous Work on Image Classification

Early Approaches

Traditional image classification techniques often relied on handcrafted features and shallow classifiers. Methods like histogram-based approaches, template matching, and edge detection were common.

Introduction of Machine Learning

With the rise of machine learning, approaches such as Support Vector Machines (SVMs) and k-Nearest Neighbors (k-NN) became popular for image classification.These methods relied on feature extraction followed by classification using these extracted features.

Transfer Learning

Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task. It leverages the knowledge gained while solving the first task to improve learning and performance on the second task. This approach is particularly useful when the amount of labeled data for the second task is limited, as it allows the model to generalize better from the larger labeled dataset of the first task. By applying transfer learning, models can achieve better performance with less labeled data, reduce training time, and generalize well to new tasks or domains.

1.VGG (Visual Geometry Group):

VGG is one of the early models of deep neural networks (DNN) used in the field of image classification and recognition. It is remarkable for its depth and complexity[7].

2.ResNet (Residual Network):

ResNet uses an innovative architecture known as ”skipping connections”. It is designed to reduce loss fading and overfitting problems that occur in deep networks[8].

3.Inception (GoogLeNet):

Inception has a complex architecture in which var- ious kernel dimensions are processed in parallel. This increases the efficiency of the model while reducing both the number of calculations and parameters[9].

4.MobileNet:

MobileNet is known for its lightweight and portable structure. It is optimized for use on mobile and embedded devices, but its performance is still high[10].

5.DenseNet:

DenseNet uses a densely connected structure by adding connections from each unit in each layer to all previous units. This improves the model's information flow and increases parameter efficiency[11].

6.EfficientNet:

EfficientNet is a family of models that are balanced in depth, width, and resolution. It is optimized for both computational and parameter efficiency[12].

7.BERT (Bidirectional Encoder Representations from Transformers):

BERT is a language model based on the attention mechanism. It is widely used and has produced groundbreaking results in the field of natural language processing[13].

8.GPT (Generative Pre-trained Transformer):

GPT is based on a Transformer architecture used to create large-scale language models. It is often used in text generation and natural language understanding tasks[14].

9.Xception:

Xception is a model built on deep neural network architecture. It is based on the ability to separate depth convolutions, resulting in a more effective model[15].

10.NASNet (Neural Architecture Search Network):

NASNet uses an architecture derivation algorithm, an instance-based learning approach used for automatic model architecture search[16].

Attention Mechanisms

Attention mechanisms have been integrated into CNN architectures to focus on relevant parts of an image, improving performance.

Notable works include ”Attentional Neural Network for Image Classification” by Yang et al. and ”Squeeze-and Excitation Networks” by Jie Hu et al. [17].

Data Augmentation and Regularization

Techniques such as data augmentation (e.g., rotation, flipping, scaling) and regularization (e.g., dropout, batch normalization) have been widely used[18].

These methods improve the generalization of image classification models and reduce overfitting[19].

Ensemble Methods

Ensemble methods, which combine predictions from multiple models, have been used to further boost performance[20].

Techniques like bagging, boosting, and stacking have been applied in the con- text of image classification[20].

These are just a few examples of the vast body of work in image classification. Research in this field continues to evolve, with ongoing efforts to improve the

accuracy, efficiency, and interpretability of image classification models

2.2.1.Image Classification in the Spatial Domain

Spatial domain refers to an area where an image is analyzed based on the locations and values of its pixels. Image classification aims to assign images to different classes based on the locations and values of these pixels.

Spatial domain image classification is a method particularly reliant on the pixel values of images. In this method, the position and value of each pixel in the image are represented with specific features. These features could include visual characteristics such as color, brightness, edges, and textures.

Image classification in the spatial domain refers to the process of categorizing or labeling images based on the spatial arrangement of pixels and the visual features present within the images themselves, without directly transforming the image into a frequency domain representation like the Fourier or Wavelet domains. In spatial domain image classification:

a)Feature Extraction:

Features are extracted directly from the raw pixel values of the image. These features may include color histograms, texture descriptors, shape information, or local image descriptors such as SIFT (Scale-Invariant Feature Transform) or SURF (Speeded-Up Robust Features).

b)Classification:

Once the features are extracted, a classifier is trained to learn patterns in the feature space and make predictions. Common classifiers used in spatial domain image classification include Support Vector Machines (SVMs), k-Nearest Neighbors (k-NN), Decision Trees, Random Forests, and Neural Net- works.

c)Evaluation:

The performance of the classifier is evaluated using metrics such as accuracy, precision, recall, F1-score, and confusion matrix on a separate test dataset.During the image classification process, a model is created using machine learning or similar techniques to differentiate pixels belonging to different classes based on these features.

This model ensures the accurate assignment of features to the correct classes, thereby enabling the classification of the image.

d)Object Recognition:

Identifying objects or patterns within images.

e)Scene Understanding:

Categorizing scenes based on their content, such as indoor vs. outdoor scenes, landscapes, or urban environments.

f)Medical Imaging:

Diagnosing diseases or abnormalities in medical images such as X-rays, MRIs, or CT scans.

g)Remote Sensing:

Analyzing satellite or aerial images for land cover classification, crop monitoring, urban planning, and environmental monitoring.

h)Security and Surveillance:

Identifying and tracking objects or individuals in surveillance videos or images.

Spatial domain image classification provides an important tool for identifying objects and features contained in an image.Spatial domain image classification can be computationally efficient and straight- forward to implement, especially for smaller datasets or when interpretability of the classification process is crucial. However, it may not capture complex frequency- based features present in the images, which can limit its performance compared to techniques that operate in the frequency domain, such as Fourier or Wavelet trans- forms followed by classification.

2.2.2.Image Classification in the Frequency Domain

Frequency domain involves analyzing an image by processing it with the help of a transformation such as the Fourier transform and transforming it from the spatial domain to the frequency domain. Image classification is analyzed in the frequency domain by converting the pixel values in the image into frequency components. In this

space, images are represented in terms of frequency components rather than raw pixel values. An overview of image classification in the frequency domain:

a)Frequency Domain Representation:

Fourier Transform is applied to separate the image into its constituent frequency components. These transformations provide a representation of the image in terms of frequency amplitudes and phases.

b)Feature Extraction:

Features are extracted from the frequency domain representation of the image. These features may include statistics of frequency components, energy distribution across different frequency bands, or other features derived from the transformed image.

c)Classification:

After features are extracted, a classifier is trained to learn pat- terns in the feature space and make predictions. Common classifiers used in frequency domain image classification include Support Vector Machines (SVMs), k-Nearest Neighbors (k-NN), Decision Trees, Random Forests, and Neural Networks.

d)Evaluation:

The performance of the classifier is evaluated using metrics such as accuracy, precision, recall, F1 score, and confusion matrix on a separate test dataset.

This approach performs classification by taking into account the distribution of frequency components in the image. This method is especially useful in cases where certain frequency components in images are important, such as texture, pattern and structure. For example, in cases where a particular pattern or structural feature is concentrated in a certain frequency range, classification in the frequency domain may be more effective.

Image classification in the frequency domain can offer several advantages:

a)Capturing Frequency Proposed Models:

By analyzing images in the frequency domain, it becomes possible to capture frequency-based patterns that may not be as obvious in the spatial domain. This can be particularly useful for tasks where frequency properties are important, such as

texture analysis or detection of periodic patterns.

b)Noise Immunity:

Frequency domain representations can often be more robust to noise, as noise can be concentrated in certain frequency ranges that can be filtered or attenuated.

c)Feature Compression:

Transforming images into the frequency domain can sometimes result in a more compact representation of image features; This can be beneficial in reducing the dimensionality of the feature space and improving computational efficiency.

However, image classification in the frequency domain also has its challenges:

d)Complexity of Transformations:

Fourier transforms can be computationally intensive, especially for large images or datasets. Additionally, choosing appropriate transformation parameters and handling boundary effects may not be trivial.

e)Interpretability:

Features extracted from the frequency domain may not al- ways be as interpretable as features extracted directly from the spatial domain. It can be difficult to understand the relevance of certain frequency components to image content.

Image classification in the frequency domain is usually performed using Fourier transform, Gabor filters or other frequency domain processing techniques. This method is especially used in areas such as pattern recognition, signal processing and image analysis.

CHAPTER 3

METHODOLOGY

Neural networks, inspired by the structure and function of the human brain, are a type of machine learning model. They consist of interconnected nodes called neurons, organized into layers. Each neuron receives inputs, performs computations, and passes its output to other neurons in the network.

Neural networks are trained using backpropagation, a process where the model adjusts its internal parameters (weights and biases) based on the error between its predictions and the actual target values. This iterative learning process enables neural networks to learn complex patterns and relationships in data, making them powerful tools for tasks such as classification, regression, and pattern recognition.

There are various types of neural networks, each designed for specific tasks and architectures. Examples include feedforward neural networks, convolutional neural networks (CNNs) for image processing, recurrent neural networks (RNNs) for sequence data, and more advanced architectures like transformers for natural language processing. In this study, Convolutional Neural Networks (CNNs) model will be discussed. Detailed information about the Convolutional Neural Networks (CNNs) algorithm will be explained in section 3.1.

3.1.Convolutional Neural Network (CNN)

A Convolutional Neural Network (CNN) is a class of deep neural networks, most commonly applied to analyzing visual imagery[21][22]. They are particularly useful for tasks such as image recognition and classification. CNNs are designed to automatically and adaptively learn spatial hierarchies of features from input data. They employ convolutional layers that apply convolution operations to the input data,

enabling the network to efficiently learn from spatial hierarchies. CNNs have been widely used in various fields, including computer vision, medical image analysis, and natural language processing.[23]

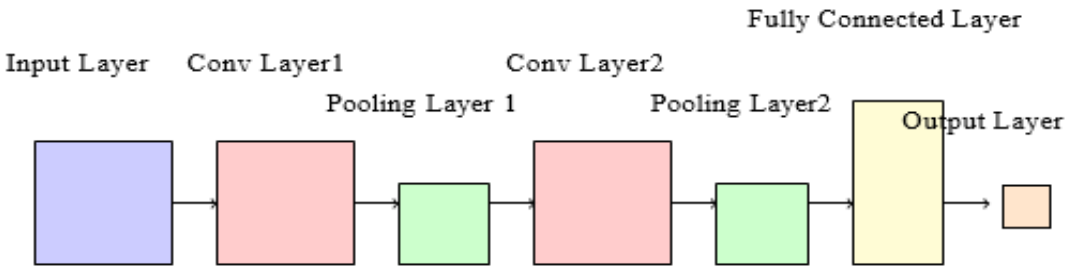

Figure 3.1: A simple representation of a Convolutional Neural Network (CNN) architecture.

3.1.1.Input Data

In Convolutional Neural Networks (CNNs), the input data typically comprises images or image-like datasets. CNNs are widely applied in tasks such as image classification, object recognition, and face detection.

The input data usually consists of images with one or more color channels. For instance, RGB images have three color channels per pixel: red, green, and blue, while grayscale images have only one channel.

The input for a CNN is typically presented as a matrix or tensor with uniform dimensions. For example, the dimensions of an RGB image could be represented as (height, width, color channels). These matrices typically contain pixel values ranging from 0 to 255.

The input data is fed into the initial layer of the CNN, often a Convolutional Layer. This layer applies specific filters to the input image, producing feature maps. These feature maps encode visual features and are further processed in subsequent layers of the CNN.

Throughout the CNN training process, the input data may undergo preprocessing steps such as data augmentation, normalization, or other transformations. These

preprocessing steps aid in improving model generalization and mitigating overfitting risks.

To summarize, the input data for a CNN usually consists of images or similar visual data types, represented as matrices or tensors. This data serves as the primary input for the CNN, enabling it to process visual information effectively.

3.1.2.Filters and Convolution Operation

In a Convolutional Neural Network (CNN), filters and the convolution operation are pivotal for extracting features from input data, especially images. Let's delve into an in-depth explanation of filters and the convolution operation:

Filters (Kernels):

Filters, also referred to as kernels, are diminutive matrices utilized to extract particular features from input data. In image processing, filters conventionally assume small square shapes. These filters are employed on localized sections of an input image to identify patterns or features like edges, textures, or shapes. Typically, filters are of modest dimensions, such as 3x3 or 5x5 matrices. Each filter is meticulously crafted to capture distinct patterns or features by assigning weights to its individual elements. Throughout the CNN's training phase, filters are dynamically learned. The network fine-tunes the weights of filters to minimize the loss function and enhance performance on the designated task. There exist various types of filters tailored for specific purposes:

1)Edge detection filters:

Examples include Sobel, Prewitt, and Scharr filters, adept at detecting edges within images.

2)Blur filters:

Gaussian blur filters, for instance, are employed to mitigate noise and smooth images.

3)Sharpening filters:

Laplacian filters, among others, augment contrast between neighboring pixels, thereby accentuating edges.

4)Custom filters:

Custom filters: Filters can be tailored to suit the requirements of a particular task or dataset.

5)Convolution Operation:

The Convolution operation entails applying a filter to an input image by systematically sliding the filter across the entire image and computing the dot product between the filter and the corresponding input patch at each position.

6)Sliding Window Operation:

The Convolutional Layer slides the filters over the input data, computing the dot product between the filter and the input patch at each position.

a)Initially, place the filter at the top-left corner of the input image.

b)Calculate the element-wise multiplication between the filter and the input patch covered by the filter.

c)Sum up the results of the element-wise multiplication to yield a single value, representing the activation of the filter at that specific position.

d)Proceed to slide the filter to the right by one pixel and repeat steps 2-3 until the entire image has been traversed.

The resulting output of the convolution operation is termed a feature map, which accentuates regions of the input image akin to the patterns encoded by the filter.

Feature Map Generation:

At each position, the dot product results in a single value, which is then used to construct a feature map. These feature maps highlight regions of the input data that are indicative of specific patterns or features encoded by the filters.

Parameter Sharing:

In CNNs, the same set of filter weights is shared across the entire input data. This parameter sharing helps in reducing the number of trainable parameters, leading to more efficient training and better generalization.

Stride and Padding:

Stride determines the step size at which the filters move across the input data. A larger stride reduces the spatial dimensions of the output feature maps. Padding involves adding extra border pixels around the input data to preserve its spatial dimensions. Padding can be ”valid” (no padding) or ”same” (padding added to maintain the same spatial dimensions).

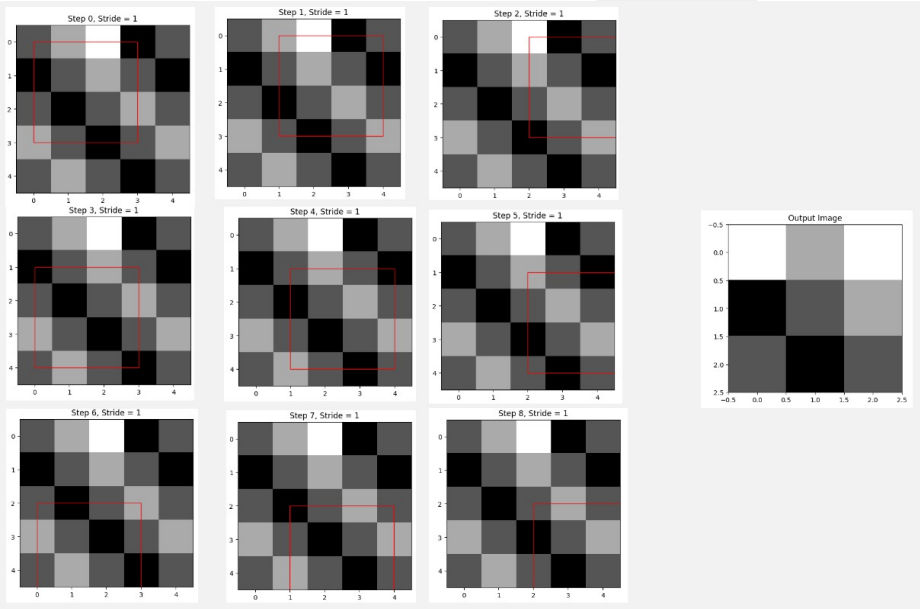

Figure 3.2: Stride

Convolutional Layer:

Within CNNs, multiple filters are concurrently applied to an input image in parallel fashion. Each filter generates its own feature map, and these individual feature maps are then stacked together to form the output of a Convolutional Layer.

Filters and the convolution operation play a crucial role across various CNN architectures, facilitating tasks such as image classification, object detection, and more. The intrinsic ability of CNNs to autonomously discern pertinent features from raw data renders them highly effective in processing visual information.

In essence, filters and the convolution operation serve as foundational elements for feature extraction within CNNs, empowering the network to acquire hierarchical representations of visual features and proficiently perform tasks related to image

analysis and recognition.

3.1.3.Activation Functions

The activation function is a crucial component of each neuron in a CNN. It introduces non-linearity into the network, allowing it to learn complex patterns and relationships within the data. The activation function determines whether a neuron should be activated based on the input it receives from the previous layer.

Non-Linearity:

One of the key characteristics of activation functions is their non-linear nature. This non-linearity is essential because many real-world datasets exhibit complex, non-linear relationships. By introducing non-linearities, the CNN can model and approximate these relationships more effectively, enabling it to capture intricate patterns in the data.

Differentiability:

Activation functions must be differentiable to facilitate the training of the CNN using techniques like backpropagation. Backpropagation involves adjusting the network's weights based on the gradients of the loss function with respect to these weights[24]. Differentiability ensures that gradients can be calculated, allowing the network to update its parameters and learn from the data.

Output Range:

The output range of an activation function determines the range of values that a neuron can produce. Different activation functions have different output ranges, which can influence the behavior and performance of the network.

Common Activation Functions in CNNs

1. ReLU (Rectified Linear Unit):

$f(x) = \max(0, x)$

ReLU is widely used due to its simplicity and effectiveness.

It replaces negative input values with zero, while positive values remain unchanged.

ReLU helps alleviate the vanishing gradient problem and accelerates con- vergence during training.

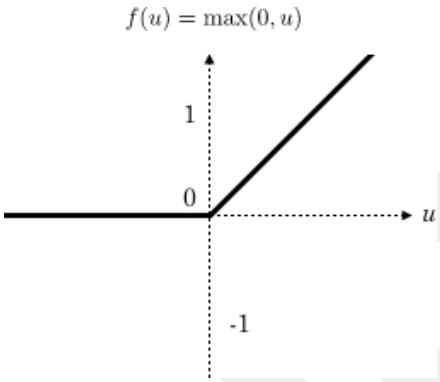

Figure 3.3 : ReLU Function Graphic

2.Sigmoid Function:

Sigmoid squashes input values to the range between 0 and 1.

It is often used in binary classification tasks to produce probability-like outputs.

However, sigmoid is prone to saturation and vanishing gradient issues, particularly for large input values.

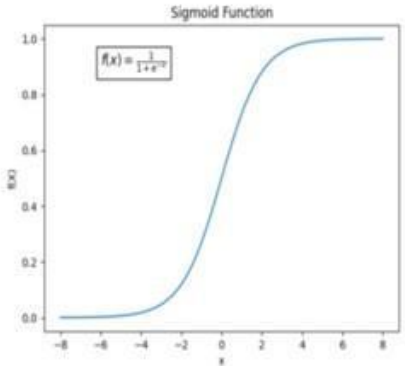

Figure 3.4: Sigmoid Function Graphic

3.Tanh (Hyperbolic Tangent) Function:

$f(x) = \tanh(x)$

Tanh squashes input values to the range between -1 and 1.

Similar to sigmoid, tanh is also used in binary classification tasks.

Tanh differs from sigmoid by producing negative values for negative inputs, resulting in a more balanced output distribution.

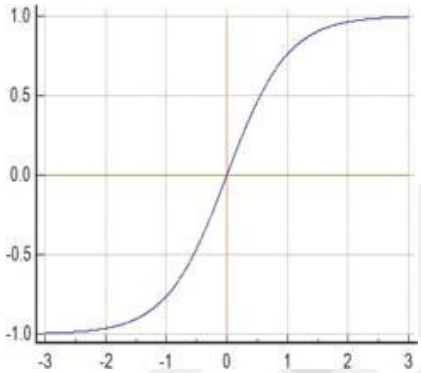

Figure 3.5: Tanh Function Graphic

4.Softmax Function:

Converts Input Values to Probabilities:

Softmax transforms an input vector into a probability distribution. Each element is between 0 and 1, and the sum of all elements is 1.

Used in Multiclass Classification:

Commonly used in multiclass classification problems to predict the class a given input belongs to.

Enhances Decision Making:

Softmax helps the model identify the class with the highest activation, improving classification accuracy.

Balances Weights:

Emphasizes the relationship between different input values. Higher input values lead to higher output probabilities.

Used in Training and Inference:

Softmax is used during both training to compute the loss and inference to make predictions.

Used with Cross-Entropy Loss:

Often used with cross-entropy loss, measuring how close the model's predictions are to true labels.

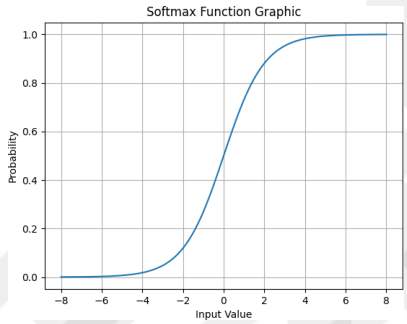

Figure 3.6: Softmax Function Graphic

In addition to the common activation functions mentioned above, there are also adaptive and specialized activation functions designed for specific tasks or architectures. Examples include Leaky ReLU, ELU (Exponential Linear Unit), and Swish.
The activation function is a critical component of Convolutional Neural Networks, enabling them to learn complex patterns and relationships in the data. Choosing the right activation function is crucial for the performance and effectiveness of the CNN, as it influences the network's ability to approximate non-linear functions and make accurate predictions.

3.1.4.Batch Normalization Operation

Batch Normalization is a technique commonly used in Convolutional Neural Networks (CNNs) to improve the training speed, stability, and performance of the model. The primary goal of Batch Normalization is to address the internal covariate shift problem during training. Internal covariate shift refers to the change in the distribution of network activations as the parameters of the preceding layers change during

training. This shift can slow down the training process and make it difficult to optimize the model effectively.

1.Normalization:

Batch Normalization normalizes the activations of each layer by subtracting the batch mean and dividing by the batch standard deviation.

This normalization step helps stabilize the training process by ensuring that the inputs to each layer have a consistent distribution, which can accelerate convergence and improve the model's ability to generalize

2.Learnable Parameters:

Batch Normalization introduces two learnable parameters per activation channel: scale (γ) and shift (β).

These parameters allow the network to learn the optimal scaling and shifting of the normalized activations, providing additional flexibility and expressiveness to the model.

Operation:

1.Mini-Batch Statistics:

During training, Batch Normalization computes the mean and standard deviation of each activation channel within a mini-batch. These statistics are used to normalize the activations of that mini-batch.

2.Normalization:

The activations are normalized by subtracting the batch mean and dividing by the batch standard deviation.

This step ensures that the activations have a mean of approximately zero and a standard deviation of approximately one, which helps stabilize the training process.

3.Scale and Shift:

After normalization, the activations are scaled and shifted using the learnable parameters (γ and β).These parameters allow the model to learn the optimal transformation of the normalized activations, preserving the representational power of the network.

4.Backpropagation:

During backpropagation, gradients are computed with respect to the normalized activations, scale, and shift parameters[24].This allows the model to learn the appropriate adjustments to the activations and parameters to minimize the loss function.

Benefits:

1.Improved Training Stability:

Batch Normalization reduces the internal covariate shift, leading to more stable training dynamics and faster convergence.

2.Regularization:

Batch Normalization acts as a form of regularization by adding noise to the activations, similar to dropout, which can help prevent overfitting.

3.Efficient Training:

Batch Normalization allows for higher learning rates, which can accelerate the training process and reduce the number of training iterations needed to achieve convergence.

4.Generalization:

By normalizing the activations, Batch Normalization helps the model generalize better to unseen data, leading to improved performance on test datasets.

Batch Normalization is a powerful technique for improving the training speed, stability, and performance of Convolutional Neural Networks. By normalizing the activations and introducing learnable parameters for scaling and shifting, Batch Normalization helps address the internal covariate shift problem and enables more efficient and effective training of deep neural networks.

3.1.5.Pooling Operation

Pooling is a fundamental operation in Convolutional Neural Networks (CNNs) used to reduce the spatial dimensions of feature maps while retaining important information. Pooling serves two primary purposes in CNNs:

a)Dimensionality Reduction:

By reducing the spatial dimensions of the feature maps, pooling reduces the computational complexity of subsequent layers and helps prevent overfitting.

b)Translation Invariance:

Pooling creates feature maps that are invariant to small translations in the input, making the network more robust to variations in the input data.

There are two main types of pooling operations commonly used in CNNs:

1.Max Pooling:

Max pooling extracts the maximum value from each patch of the input feature map.

It retains the most important features while discarding less relevant in- formation.

Max pooling is often preferred due to its simplicity and effectiveness.

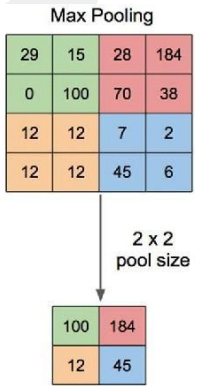

Figure 3.7: Max Pooling

2.Average Pooling:

Average pooling computes the average value of each patch of the input feature map. It smoothes the feature maps and reduces the impact of noisy activations.

Average pooling may be used in scenarios where preserving the mean intensity of features is important.

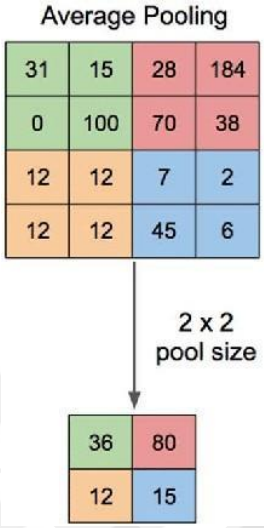

Figure 3.8: Average Pooling

There are 3 hyperparameters. These are pooling size, stride and padding.

Pooling is typically applied after convolutional layers in CNNs to progressively reduce the spatial dimensions of the feature maps while retaining important information.

Pooling is a critical operation in CNNs for reducing the spatial dimensions of feature maps and extracting important features from the input data. By downsampling the feature maps and introducing translation invariance, pooling helps CNNs learn hierarchical representations of the input data and improve their ability to generalize to unseen samples.

3.1.6.Dropout Operation

Dropout is a regularization technique commonly used in Convolutional Neural Networks (CNNs) to prevent overfitting and improve the generalization capability of the model.

The primary purpose of Dropout is to reduce the interdependency between neurons within the network by randomly deactivating a certain percentage of neurons during each training iteration. This prevents the network from relying too heavily on any specific set of features and encourages robustness in the learned features.

Dropout is a powerful regularization technique used in CNNs to prevent overfitting and improve the generalization capability of the model. By randomly deactivating neurons during training, Dropout encourages the network to learn more robust and generalizable features, leading to better performance on unseen data. It is widely used in practice and has been shown to be effective in improving the performance of deep neural networks.

3.1.7.Fully Connected Layers

Fully Connected Layers, also known as dense layers, are a crucial component of Convolutional Neural Networks (CNNs) used for tasks like image classification, object detection, and more.

Fully Connected Layers are responsible for learning high-level features from the output of convolutional and pooling layers and making predictions based on these features. They perform classification or regression tasks by taking the flattened output of the preceding layers and mapping it to the desired output size (e.g., class probabilities).

It consists of flattening, weighted sum and activation and output prediction operations. During training, the parameters (weights and biases) of the fully connected layers are optimized using backpropagation and gradient descent[24]. The loss between the predicted output and the ground truth is minimized iteratively by adjusting the weights of the network.

Fully Connected Layers are commonly used in the final stages of CNN architectures for tasks like image classification, where they map the learned features to class labels. They can also be used in other tasks like regression, where they predict continuous values.

Fully Connected Layers play a vital role in CNNs by learning high-level features from

the output of convolutional and pooling layers and making predictions based on these features. They provide the network with the capability to perform complex tasks like image classification and regression, making them a fundamental component of modern deep learning architectures.

3.1.8.Loss Function and Optimization

The loss function and optimization are essential components of training Convolutional Neural Networks (CNNs).

Loss Function: The loss function measures the difference between the predicted output of the network and the actual target labels. It quantifies how well the model is performing on the training data and provides feedback for adjusting the model's parameters during training. Various loss functions can be used depending on the nature of the task:

1.Cross-Entropy Loss (or Log Loss):

Commonly used for classification tasks, especially when dealing with multiple classes. Measures the dissimilarity between the predicted probability distribution and the true distribution of class labels.

Encourages the model to assign higher probabilities to the correct class labels.

2.Mean Squared Error (MSE):

Typically used for regression tasks.

Measures the average squared difference between the predicted and actual target values.

Penalizes large errors more heavily than smaller ones.

3.Binary Cross-Entropy Loss:

Specifically used for binary classification tasks.

Similar to cross-entropy loss but adapted for binary classification problems with only two classes.

4.Custom Loss Functions:

Tailored loss functions can be designed for specific tasks or to address particular challenges in the data.

Optimization: Optimization algorithms are used to minimize the loss function by adjusting the parameters (weights and biases) of the neural network during training. The goal is to find the set of parameters that result in the lowest possible loss value on the training data. Common optimization algorithms include:

a)Stochastic Gradient Descent (SGD):

The most basic optimization algorithm.

Updates the parameters in the direction that reduces the loss, based on the gradient of the loss function with respect to each parameter.

Can suffer from slow convergence and oscillations, especially in high dimensional spaces.

b) Adam (Adaptive Moment Estimation):

An adaptive learning rate optimization algorithm[25].

Maintains separate learning rates for each parameter and adapts them based on the past gradients.

Combines the advantages of both AdaGrad and RMSProp optimization algorithms.

c)RMSProp (Root Mean Square Propagation):

Another adaptive learning rate optimization algorithm[25].

Divides the learning rate by an exponentially decaying average of squared gradients to scale the updates.

d)Adagrad (Adaptive Gradient Algorithm):

Adjusts the learning rate for each parameter based on the frequency of updates.

It accumulates the squared gradients for each parameter and uses them to scale the learning rate.

The loss function quantifies the model's performance during training, while optimization algorithms adjust the model's parameters to minimize this loss. Proper selection of loss function, optimization algorithm, and hyperparameters is critical for training CNNs effectively and achieving optimal performance on the desired tasks.

3.2 Proposed Model

In this study, the CNN algorithm is used for image classification. The algorithm consists of 4 layers. The kernel size is set to 3x3, "same" padding is applied, and "ReLu" activation function is used. Additionally, Batch Normalization, Dropout, and Max Pooling are employed. In the fully connected layer, the "softmax function" is used as the activation function. "Sparse categorical crossentropy" is utilized as the loss function, and "Adam" is chosen as the optimizer.

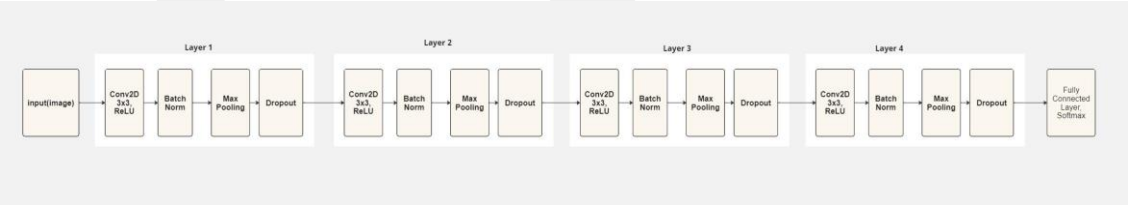

Figure 3.9 : Proposed Model

CHAPTER 4

EXPERIMENTS

The aim of this study was to investigate whether the image can be used by taking the Fourier Transform of the input images for image classification with convolutional neural networks. Models were created for both RGB images and Fourier spectra. Models trained and evaluated with RGB images were then compared with models trained via Fourier Transform. Additionally, various applications were made on the image while adjusting the data set. It will be explained in more detail in 4.1.

The experiments were carried out on a laptop and the Python programming lan- guage was used. Multiple open source Python libraries were used: one of the most important was Tensorflow. It is an open-source software library for machine learning tasks developed by the Google Brain Team that offers an API that allows it to be used with a variety of programming languages. TensorFlow provides, among other things, methods for creating neural networks that follow desired architectures, as well as methods for training and evaluating them. Keras was used as the interface to the TensorFlow library. Finally, NumPy was used to perform mathematical opera- tions on multidimensional matrices. The training process was explained in section 4.2. The results are explained in section 4.3.

4.1.Data Sets

The Convolutional Neural Network (CNN) algorithm employed in this thesis follows the model described in Section 3.3. Training was conducted utilizing the CPU of the device in use, imposing limitations on both calculation complexity and the number of epochs.

Datasets utilized include CIFAR-10, MNIST Fashion, and MNIST Digit. The CIFAR-

10 dataset comprises 60,000 images distributed across 10 classes, with images sized at 32x32x3, denoting RGB color channels. Conversely, both MNIST Fashion and MNIST Digit datasets encompass 60,000 grayscale images at a size of 28x28.

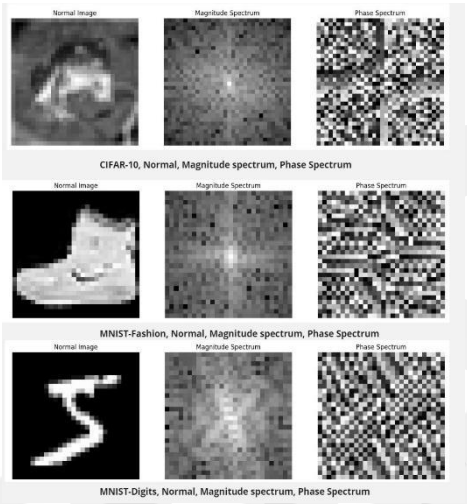

Figure 4.1 : CIFAR-10, MNIST-Fashion, MNIST-Digits

Data loading was facilitated using Keras. Subsequently, the datasets were partitioned into training, validation, and testing subsets.

Figure 4.1 displays sample RGB images from the CIFAR-10 dataset utilized for training, testing, and validation.

Figure 4.2 illustrates sample images from the MNIST-Digits dataset employed for training, testing, and validation.

Figure 4.3 showcases sample images from the MNIST-Fashion dataset used for training, testing, and validation.

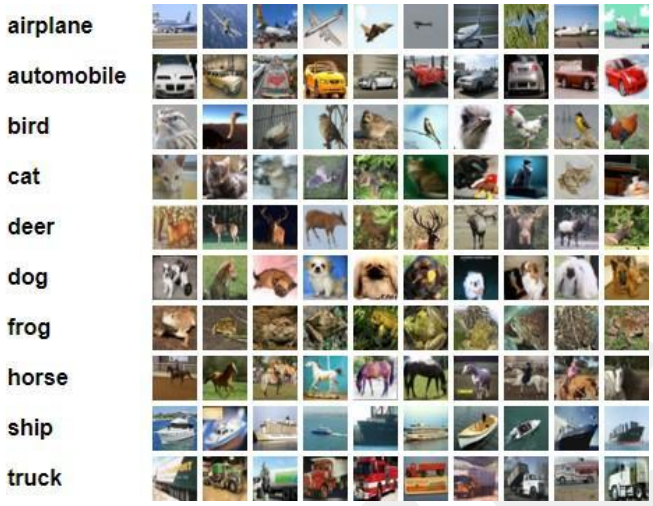

Figure 4.2 : Example images from the CIFAR-10 dataset classes

Figure 4.3: Example images from the MNIST-Digits dataset classes.

4.2.Pre-processing

In the CNN model, preprocessing steps were meticulously applied to each 32x32x3 (RGB) image in the CIFAR-10 dataset with the aim of optimizing processing load and enhancing feature extraction capabilities.

Initially, the RGB images underwent a grayscale conversion, reducing them to a manageable 32x32 size to facilitate subsequent processing steps.

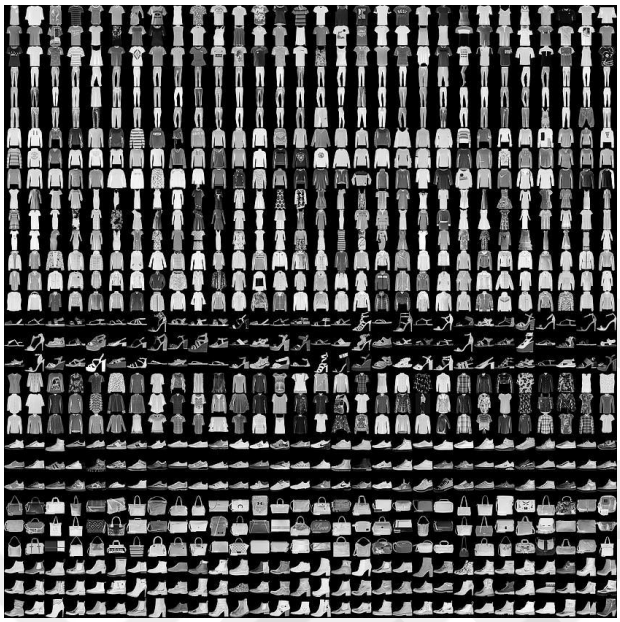

Figure 4.4 : Example images from the MNIST-Fashion dataset classes.

Following this, each 32x32 image was divided into smaller 2x2 subimages. This subdivision process occurred across the entire row, with a stride value of 1, traversing from the zeroth index to the last index. Upon reaching the end of a row, the process was repeated in the subsequent row. This iterative process was performed across all rows and columns, resulting in a 50 percent overlap due to the stride value of 1. The purpose of this overlapping was to investigate potential correlations between adjacent data points(shown in figure 4.5 and 4.6). Additionally, 2x2 subimages were extracted again, this time with a shift process that left a gap between indexes to prevent overlap(Figure 4.7). Padding was applied throughout these operations to prevent loss of information at the leading and trailing indexes. This meticulous approach aimed to capture intricate details and patterns present in the images.

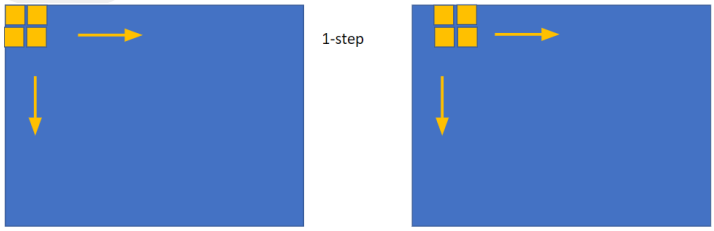

Figure 4.5 : with overlap

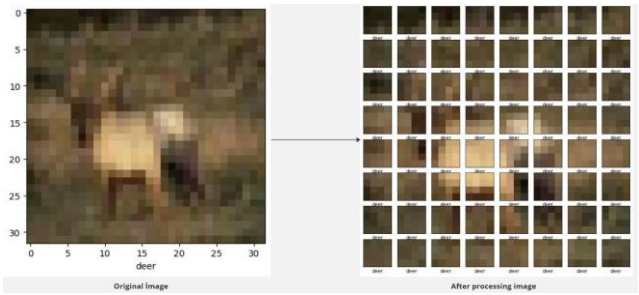

Figure 4.6 : Original image and after processing

Subsequently, a new dataset was generated by subjecting each of these small subimages to the Fourier Transform.This transformation enabled the extraction of frequency-based features from the images.

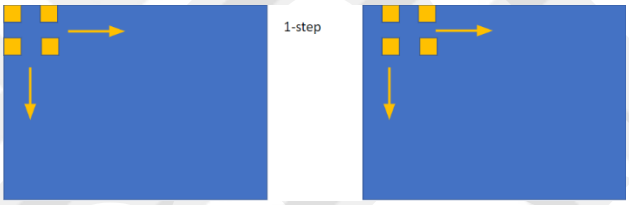

Figure 4.7 : without overlap

Upon completion of the Fourier Transform, the dimensions of the new dataset were determined. To mitigate potential scaling issues, the data underwent scaling by applying the logarithm to each image.

Following these preprocessing steps, the small 2x2 subimages were reassembled to restore them to their original 32x32 dimensions. This reconstruction process ensured that the fundamental features extracted from the images remained intact.

These comprehensive preprocessing steps played a pivotal role in enhancing the model's capability to extract meaningful features from the CIFAR-10 dataset, thereby contributing to the overall improvement in the classification performance of the CNN model.

Furthermore, all operations except grayscale conversion were applied to the MNIST-Digits and MNIST-Fashion datasets, ensuring consistency in preprocessing across different datasets.

4.3.Training the Models

4.3.1.Architectural Design and Classification Processes

The architectural design and classification processes used in this study are detailed in Section 3.2. Initially, a basic model was applied for RGB image classification using the CIFAR-10 dataset. The same model was also applied to the MNIST-Digits and MNIST-Fashion datasets. The models were trained for 1000 epochs each. To prevent overfitting, the early stopping method was implemented. This method effectively prevented the model from overfitting.

4.3.2.Application of the Basic Model and Training Process

Following the application of the basic model, a classification model specific to the application studied was developed based on the same basic model. In this context, four different models were trained for the CIFAR-10, MNIST-Digits, and MNIST- Fashion datasets. For the CIFAR-10 dataset, one of the trained models performed the sliding operation with a standard stride value of 1, while the other models performed the sliding operation with 50% and 0% overlap, respectively. The same processes were repeated for the MNIST-Digits and MNIST-Fashion datasets.

4.3.3.Training Duration and Performance of the Models

Training the models for 1000 epochs increased the training duration but resulted in more reliable classification outcomes. The trained models will provide benchmark results for the experiments conducted.

4.3.4.Analysis of Training and Validation Results

Figures 4.8, 4.9, and 4.10 show the Training and Validation Accuracy values and Training and Validation Loss values for the CNN model throughout the training process on the CIFAR-10, MNIST-Digits, and MNIST-Fashion datasets. These figures indicate that both Training and Validation Loss values decreased, suggesting that

the model effectively learned the data.

Figures 4.11, 4.12, and 4.13 present the results of the Fourier Transform. These figures also show that, despite some occasional spikes, both Training and Validation Loss values generally decreased. Similarly, the Training and Validation Accuracy values were observed to increase.

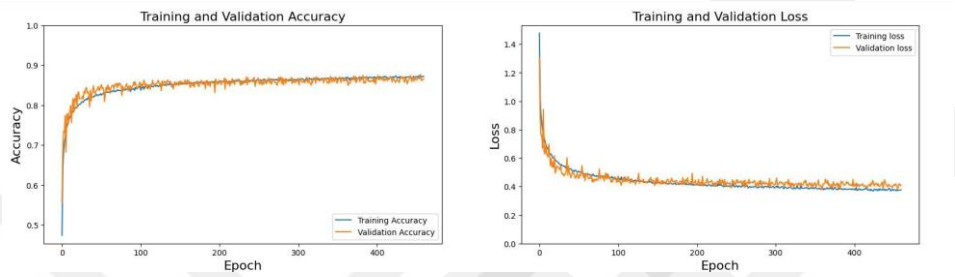

Figure 4.8 : Training and Validation results for CIFAR-10 dataset

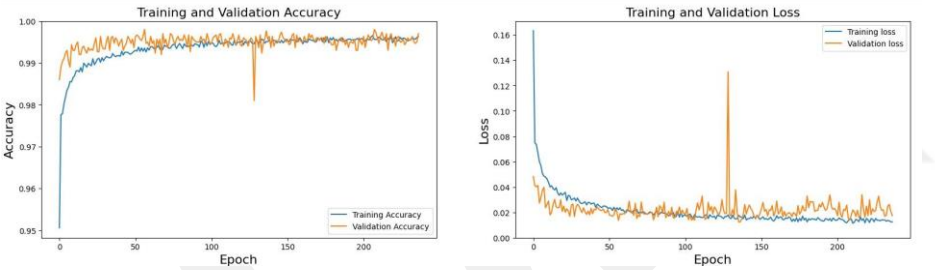

Figure 4.9 : Training and Validation results for MNIST-Digits dataset

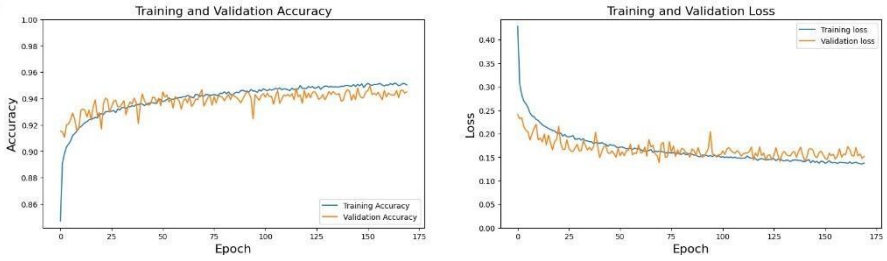

Figure 4.10 : Training and Validation results for MNIST-Fashion dataset

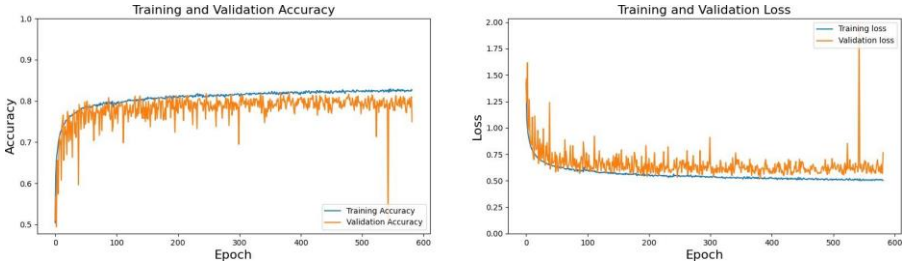

Figure 4.11 : Fourier Transform results for CIFAR-10 dataset

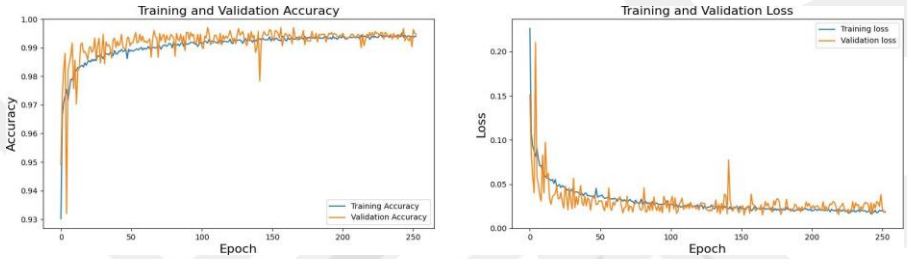

Figure 4.12 : Fourier Transform results for MNIST-Digits dataset

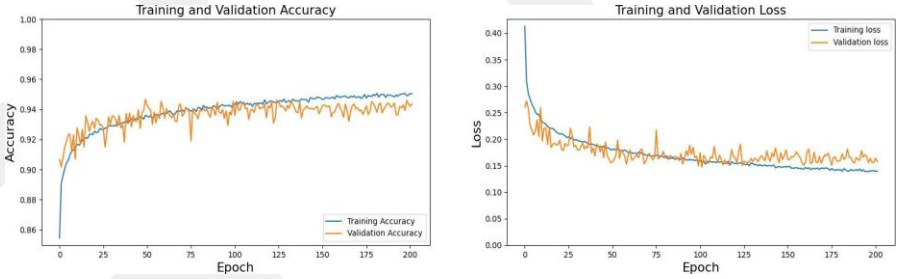

Figure 4.13 : Fourier Transform results for MNIST-Fashion dataset

These findings demonstrate that the model effectively learned and that the early stopping method was effective in preventing overfitting.

4.4.Results and Analysis

This study is designed to examine the effectiveness of CNN models in image classification in the frequency domain. Model architectures and training processes were evaluated on various datasets such as CIFAR-10, MNIST-Digits, and MNIST-Fashion. The loss values used during training indicate how reliably the model adapts to the data.

The loss values obtained during training in Figures 4.8, 4.9, and 4.10 demonstrate that the selected model adapts more reliably to the data. However, this reliability does not reflect in the classification results on the test data. The accuracy values in Table 1 indicate that the model's ability to classify test data is weaker than the model discussed in the thesis.

CNNs were observed to be more suitable for image classification in the time do- main. This model architecture achieved better performance in image classification in the time domain.

The results obtained from classifying input images obtained through Fourier transformation did not outperform those obtained in the time domain classification. This indicates that Fourier transformation may lead to information loss in some cases and adversely affect classification performance.

In conclusion, this study evaluated the effectiveness of CNN models in image classification in the time domain. The results demonstrate that image classification in the time domain is more successful compared to frequency domain approaches such as Fourier transformation. However, more comprehensive studies on different datasets and model architectures will contribute to a deeper understanding of these results.

Table 4.1: Accuracy values for different datasets and domains

Dataset	Spatial Domain	Original Frequency Domain	Frequency Domain
CIFAR-10	0.8593	0.4771	0.7865
MNIST-Digits	0.9946	0.9418	0.9326
MNIST-Fashion	0.9341	0.8362	0.8752

CHAPTER 5

CONCLUSION

This thesis explored the effectiveness of utilizing Fast Fourier Transform (FFT) transformed images as inputs for Convolutional Neural Networks (CNNs) in image classification tasks. Various CNN architectures and training methodologies were assessed using benchmark datasets, including CIFAR-10, MNIST-Digits, and MNIST- Fashion. Training loss values, as depicted in Figures 4.8, 4.9, and 4.10, showed that the models adapted well to the data. However, this adaptation did not necessarily result in improved classification accuracy on the test data, as evidenced by the accuracy values presented in Table 1.

The research revealed that CNNs perform more effectively in the spatial (time) domain compared to the frequency domain. Models trained on spatial domain images consistently achieved better results than those trained on FFT-transformed images. This suggests that essential information for accurate classification may be lost during the Fourier transformation, adversely affecting the model's performance.

Although the frequency domain approach offered some valuable insights into data representation, it did not surpass the traditional spatial domain approach in terms of results. This underscores the robustness of spatial domain data for CNN-based image classification tasks. The observed performance gap between the two domains indicates that further refinement and the development of hybrid approaches might be required to fully harness frequency domain information.

In conclusion, this study demonstrates that image classification with CNNs is more successful in the spatial domain than in the frequency domain. While FFT provides an

alternative perspective on image data, it may introduce challenges that can hinder classification performance. Future research should focus on developing hybrid models that incorporate both spatial and frequency domain information, as well as examining the effects of FFT on a broader range of datasets and more advanced CNN architectures. Such efforts will enhance our understanding of the strengths and limitations of frequency domain analysis in deep learning.

The findings of this thesis contribute to the field of computer vision by providing insights into the use of frequency domain techniques in CNN workflows. Future studies are encouraged to validate these findings and explore innovative methodologies to improve image classification performance by effectively combining spatial and frequency domain data.

REFERENCES

[1] A. D. Akwaboah, "Implementation of Convolutional Neural Networks for CIFAR-10 Image Classification", https://www.researchgate.net/publication/337240963_Convolutional_Neural_Network_for_CIFAR-10_Dataset_Image_Classification, November 2019.

[2] S. Ajala, "Convolutional Neural Network Implementation for Image Classification using CIFAR-10", https://www.researchgate.net/publication/355972159_Convolutional_Neural_Network_Implementation_for_Image_Classification_using_CIFAR-10_Dataset, November 2021.

[3] H. Pan "Learning Convolutional Neural Networks in Frequency Domain", https://www.researchgate.net/publication/359971778_Learning_Convolutional_Neural_Networks_in_Frequency_Domain, April 2022.

[4] S. Totterstrom, "Frequency Domain Image Classification with Convolutional Neural Networks" Bachelor's Thesis, Tampere University, 2023.

[5] Gonzalez, R. C., & Woods, R. E., 2020, *Digital Image Processing* (3th ed.) [Print], Vol.1.,https://dl.ebooksworld.ir/motoman/Digital.Image.Processing.3rd.Edition.www.EBooksWorld.ir.pdf.

[6] J. W. Cooley, and J. W. Tukey "An Algorithm for the Machine Calculation of Complex Fourier Series", *Mathematics of Computation*, vol. 19(90) pp. 297-301 1965.

[7] K. Simonyan, and A. Zisserman "Very Deep Convolutional Networks for Large-Scale Image Recognition", https://arxiv.org/abs/1409.1556, 10 April 2015.

[8] K. He, X. Zhang, S. Ren, and J. Sun "Deep Residual Learning for Image Recognition", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770-778 2016.

[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, and D. Anguelov, "Going Deeper with Convolutions", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-9 2015.

[10] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, and H. Adam, "Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications", https://arxiv.org/abs/1704.04861 , 17 April 2017.

[11] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700-4708 2017.

[12] M. Tan, and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks", *International Conference on Machine Learning*, pp. 6105-6114 2019.

[13] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", https://arxiv.org/abs/1810.04805, 24 May 2019

[14] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners", *OpenAI Blog* ,vol. 1(8) pp. 9 2019.

[15] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1251-1258 2017.

[16] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning Transferable Architectures for Scalable Image Recognition", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8697-8710 2018.

[17] J. Hu, L. Shen, and G. Sun, "Squeeze and Excitation Networks", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7132-7141 2018.

[18] C. Shorten, T. M. Khoshgoftaar, and T. R. Mercer, "Data Augmentation: A Survey", *ACM Computing Surveys (CSUR)* , vol. 52(1) pp. 1-38 2019.

[19] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple Way to Prevent Neural Networks from Overfitting*"*, *The Journal of Machine Learning Research*, vol. 15(1) pp. 1929-1958 2014.

[20] L. K. Hansen, and P. Salamon, "Neural Network Ensembles ", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12(10) pp. 993-1001 1990

[21] M. D. Zeiler, and R. Fergus, "Visualizing and Understanding Convolutional Networks ", *European Conference on Computer Vision*, pp. 818-833 2014.

[22] C. Li, M. Wand, S. Bao, and S. Fidler, "Visualizing and Understanding Convolutional Networks", *European Conference on Computer Vision*, pp. 818-833 2016.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", *Advances in Neural Information Processing Systems*, vol. 25 pp. 1097-1105 2012.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition*"*, *Proceedings of the IEEE*, vol. 86(11) pp. 2278-2324 1998.

[25] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio Deep Learning (Adaptive Computation and Machine Learning series). Cambridge, MA: MIT Press 2016.