



AFRL-RY-WP-TR-2022-0185

OPTIMIZATION OF CONVOLUTIONAL NEURAL NETWORKS FOR ENHANCED COMPRESSION TECHNIQUES AND COMPUTER VISION APPLICATIONS

Christopher Javier Couture Del Valle
University of Dayton

JUNE 2022
Thesis

DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.

See additional restrictions described on inside pages

© 2022 Christopher Javier Couture Del Valle. All rights reserved.

STINFO COPY

**AIR FORCE RESEARCH LABORATORY
SENSORS DIRECTORATE
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7320
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE**

REPORT DOCUMENTATION PAGE

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE June 2022		2. REPORT TYPE Thesis		3. DATES COVERED	
				START DATE 14 June 2022	END DATE 14 June 2022
4. TITLE AND SUBTITLE OPTIMIZATION OF CONVOLUTIONAL NEURAL NETWORKS FOR ENHANCED COMPRESSION TECHNIQUES AND COMPUTER VISION APPLICATIONS					
5a. CONTRACT NUMBER FA8650-18-C-1607		5b. GRANT NUMBER N/A		5c. PROGRAM ELEMENT NUMBER 63270F/63203F/35022F	
5d. PROJECT NUMBER 4335/69DF/5023		5e. TASK NUMBER N/A		5f. WORK UNIT NUMBER Y1R7	
6. AUTHOR(S) Christopher Javier Couture Del Valle					
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University of Dayton 300 College Park Dayton, OH 45469					8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory, Sensors Directorate Wright-Patterson Air Force Base, OH 45433-7320 Air Force Materiel Command, United States Air Forces				10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/RYDR	11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-RY-WP-TP-2022-0185
12. DISTRIBUTION/AVAILABILITY STATEMENT DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.					
13. SUPPLEMENTARY NOTES PAO case number AFRL-2022-2815, Clearance Date 14 June 2022. © 2022 Christopher Javier Couture Del Valle. All rights reserved. Submitted to The School of Engineering of the University of Dayton in partial fulfillment of the requirements for the Degree of Master of Science in Engineering. This work was funded in whole or in part by Department of the Air Force. The U.S. Government has for itself and others acting on its behalf an unlimited, paid-up, nonexclusive, irrevocable worldwide license to use, modify, reproduce, release, perform, display, or disclose the work by or on behalf of the U. S. Government. Report contains color.					
14. ABSTRACT Image compression algorithms are the basis of media transmission and compression in the field of image processing. Decades after their inception, algorithms such as the JPEG image codec continue to be the industry standard. A notable research topic gathering momentum in the field of compression is deep learning (DL). This paper explores the optimization of DL models for ideal image compression and object detection (OD) applications. The DL model to be optimized is based upon an existing compression framework known as the CONNECT model. This framework wraps the traditional JPEG image codec within two convolutional neural networks (CNNs). The first network, ComCNN, focuses on compressing an input image into a compact representation to be fed into the image codec. The second network, RecCNN, focuses on reconstructing the output image from the codec as similarly as possible to the original image. To enhance the performance of the CONNECT model, an optimization software called Optuna wraps the framework. Hyperparameters are selected from each CNN to be evaluated and optimized by Optuna. Once the CONNECT model produces ideal results, the output images are applied to the YOLOv5 OD network. This paper explores the impact of DL hyperparameters on image quality and compression metrics. In addition, a detection network will provide context to the effect of image compression on computer vision applications.					
15. SUBJECT TERMS data compression, hyperparameter tuning, image compression, infrared imagery, machine learning, object detection					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT SAR		18. NUMBER OF PAGES 85
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified			
19a. NAME OF RESPONSIBLE PERSON Marc Hoffman				19b. PHONE NUMBER (Include area code)	

OPTIMIZATION OF CONVOLUTIONAL NEURAL NETWORKS FOR ENHANCED
COMPRESSION TECHNIQUES AND COMPUTER VISION APPLICATIONS

Thesis

Submitted to

The School of Engineering of the
UNIVERSITY OF DAYTON

In Partial Fulfillment of the Requirements for

The Degree of

Master of Science in Engineering

By

Christopher Javier Couture Del Valle

Dayton, Ohio

August, 2022



Name: Couture Del Valle, Christopher Javier

APPROVED BY:

Bradley M. Ratliff, Ph.D.
Advisory Committee Chairman
Associate Professor, Electrical and
Computer Engineering

Eric J. Balster, Ph.D.
Committee Member
Associate Professor and Chair,
Electrical and Computer Engineering

Barath Narayanan, Ph.D.
Committee Member
Adjunct Professor, Electrical and
Computer Engineering

Robert J. Wilkens, Ph.D., P.E.
Associate Dean for Research and Innovation
Professor
School of Engineering

Eddy M. Rojas, Ph.D., M.A., P.E.
Dean, School of Engineering

© Copyright by
Christopher Javier Couture Del Valle
All rights reserved
2022

ABSTRACT

OPTIMIZATION OF CONVOLUTIONAL NEURAL NETWORKS FOR ENHANCED COMPRESSION TECHNIQUES AND COMPUTER VISION APPLICATIONS

Name: Couture Del Valle, Christopher Javier
University of Dayton

Advisor: Dr. Bradley M. Ratliff

Image compression algorithms are the basis of media transmission and compression in the field of image processing. Decades after their inception, algorithms such as the JPEG image codec continue to be the industry standard. A notable research topic gathering momentum in the field of compression is deep learning (DL). This paper explores the optimization of DL models for ideal image compression and object detection (OD) applications.

The DL model to be optimized is based upon an existing compression framework known as the CONNECT model. This framework wraps the traditional JPEG image codec within two convolutional neural networks (CNNs). The first network, ComCNN, focuses on compressing an input image into a compact representation to be fed into the image codec. The second network, RecCNN, focuses on reconstructing the output image from the codec as similarly as possible to the original image. To enhance the performance of the CONNECT model, an optimization software called Optuna wraps the framework. Hyperparameters are selected from each CNN to be evaluated and optimized by Optuna. Once the CONNECT model produces ideal results, the output images are applied to the YOLOv5 OD network.

This paper explores the impact of DL hyperparameters on image quality and compression metrics. In addition, a detection network will provide context to the effect of image compression on computer vision applications.

ACKNOWLEDGMENTS

I would first like to thank the team at the Air Force Research Lab (AFRL) for funding and supporting this work. The biweekly meetings have vastly improved my public speaking skills and confidence when presenting complex information. Marc Hoffman and Stefan Westberg have been extremely supportive in these meetings while providing the necessary feedback to be successful in this project.

A big thanks to Dr. Bradley Ratliff for introducing me to this project and supporting me as my graduate advisor. You have always been there to assist me and continue to support me as I complete this thesis. Thank you for providing me with this opportunity and welcoming me into the Applied Sensing Lab.

I would also like to thank the Scalable Computing team at the University of Dayton Research Institute (UDRI) for helping me through this process. David Fan, Jonathan Skeans and Matthew Gnacek have been my pillars of support allowing me to grow as a computer engineer. You have all been a critical piece in the growth of the CONNECT project and I cannot thank you enough for all I have learned.

Finally, I would like to thank my parents, Chris and Lymary, along with my sister Isabella for providing me with the love and support that's allowed me to get where I am today.

TABLE OF CONTENTS

ABSTRACT	iii
ACKNOWLEDGMENTS	iv
LIST OF FIGURES	vii
LIST OF TABLES	ix
CHAPTER I. INTRODUCTION	1
1.1 Related Works	8
CHAPTER II. DEEP LEARNING MODELS	9
2.1 Compression Architecture Overview	9
2.1.1 ComCNN	10
2.1.2 RecCNN	11
2.1.3 Image Codecs	12
2.1.4 Compression Architecture Details	13
2.1.5 Compression Architecture Results	15
2.2 Object Detection Network Overview	16
2.2.1 YOLOv5 Model Overview	16
2.2.2 Training and Datasets	17
2.2.3 YOLOv5 Model Results	19
2.3 General Architecture Overview	19
CHAPTER III. MODEL PERFORMANCE EVALUATION	20
3.1 Image Quality Metrics	20
3.1.1 Mean Square Error	21
3.1.2 Peak Signal to Noise Ratio	21
3.1.3 Structural Similarity Index	22
3.2 Image Compression Metrics	23
3.2.1 Bits Per Pixel	23
3.2.2 Compression Ratio	24
3.3 Object Detection and Classification Metrics	24
3.3.1 Confusion Matrix	24
3.3.2 Precision, Recall and F-1 Score	25
3.3.3 mAP Score	26
3.4 Visual Perception	28
CHAPTER IV. HYPERPARAMETER EXPLORATION	30
4.1 Optuna Framework	30
4.2 Variance and Non-determinism	33
4.3 Model Hyperparameters	33
4.3.1 Learning Rate	34
4.3.2 Number of Layers	34
4.3.3 Kernel Size and Strides	34

4.3.4	Initializer	35
4.3.5	Activation Function	35
4.3.6	Quality Factor	36
CHAPTER V. EXPERIMENTAL SETUP AND RESULTS		37
5.1	Experimental Setup	37
5.1.1	OS and Software	37
5.1.2	Hardware	38
5.1.3	Training and Testing Pipeline	39
5.2	General Model Optimization	41
5.2.1	8-bit EO	42
5.2.2	14-bit IR	46
5.2.3	8-bit IR	50
5.3	Quality Factor Analysis of Optimized Model	52
5.3.1	8-bit EO	53
5.3.2	8-bit IR	55
5.4	Object Detection Analysis	58
5.4.1	8-bit EO	58
5.4.2	8-bit IR	60
CHAPTER VI. FUTURE WORK AND CONCLUSION		64
6.1	Future Work	64
6.1.1	New Image Data Types	64
6.1.2	Modernized Image Codecs	65
6.1.3	Performance Evaluation Metrics	65
6.1.4	16-bit in Object Detection	66
6.2	Conclusion	66
BIBLIOGRAPHY		69

LIST OF FIGURES

1.1	High Level Overview of Applied AI Methods	3
1.2	Low to High Level Learned Features from a CNN	4
1.3	RGB to Grayscale Image Comparison	5
1.4	RGB to IR Image Comparison	6
2.1	CONNECT Framework Image Processing Pipeline Overview	10
2.2	ComCNN Image Downscaling Pipeline	11
2.3	RecCNN Image Reconstruction Pipeline	12
2.4	YOLOv5 Object Detection Network Pipeline Overview	17
2.5	Example of Bounding Box Object Detections in the COCO dataset	18
2.6	CONNECT Framework to YOLOv5 Network General Architecture Pipeline . .	19
3.1	Confusion Matrix Calculations for Classification Metrics	25
3.2	Visualization of the Intersection Over Union Calculation	27
3.3	CONNECT Framework Image Outputs for Visual Perception	28
4.1	High Level Optuna Optimization Process Flowchart	31
5.1	Optuna Training and YOLOv5 Network Testing Flowchart	40
5.2	8-Bit EO Model Optimization Hyperparameter Importances Plot	44
5.3	8-bit EO Visual Perception from Optimized and Original Model	45
5.4	14-Bit IR Model Optimization Hyperparameter Importances Plot	47
5.5	14-bit IR Visual Perception from Optimized Models	49
5.6	8-bit IR Visual Perception from Optimized Models	52
5.7	8-bit EO QF Analysis Testing Plots for JPEG and CONNECT	53
5.8	8-bit EO QF Analysis Testing Plot Overlay for JPEG vs CONNECT	54

5.9	8-bit EO QF Compression Plot Results for JPEG vs CONNECT	54
5.10	8-bit IR QF Resized vs Non-Resized Results for JPEG	55
5.11	8-bit IR QF Resized Model Testing Results of JPEG vs Optimized Models . .	56
5.12	8-bit IR QF Non-Resized Model Testing Results of JPEG vs Optimized Models	57
5.13	8-Bit EO Detection QF Baseline Plot	59
5.14	8-Bit EO Detection Visualization Output with Bounding Boxes	60
5.15	8-Bit IR Detection Visualization Output with Bounding Boxes	62
5.16	8-Bit IR Resized Detection QF Results of JPEG vs Optimized Models	62
5.17	8-Bit IR Non-Resized Detection QF Results of JPEG vs Optimized Models . .	63

LIST OF TABLES

5.1	8-Bit EO Model Optimization Training Results	43
5.2	8-Bit EO Optimized Model Testing Results	45
5.3	14-Bit IR Model Optimization Small Training Results	47
5.4	14-Bit IR Model Optimization Large Training Results	48
5.5	14-bit IR Optimized Model Testing Results	49
5.6	8-Bit IR Model Optimization Small Training Results	50
5.7	8-Bit IR Model Optimization Large Training Results	51
5.8	8-Bit IR Optimized Model Testing Results	51
5.9	8-Bit EO Detection and Image Metric Results Chart	59
5.10	8-Bit IR Detection and Image Metric Results Chart	61

CHAPTER I

INTRODUCTION

As imaging technologies develop rapidly, image size and quality have drastically increased. To tackle this data allocation problem, compression algorithms such as JPEG were created to reduce transmission and data storage costs across the board. These algorithms remain the industry standard even as images continue to grow larger. As a result, image processing research has shifted its focus towards utilizing modern deep learning techniques. Deep learning has seen success in a variety of notable image processing applications such as object detection and classification. The exploration of such techniques in image compression are thus warranted and the focus of this paper.

The goal of this thesis can be divided into three objectives. The first of which is the optimization of an existing framework, the Convolutional Neural Networks and Enhanced Compression Techniques Framework (CONNECT). Optimization involves determining the ideal hyperparameters in its convolutional neural networks. The success of optimization is determined by the quality and compression metrics yielded by output images of different types, notably gray-scale and IR. The second objective is comparing the outputs of the framework to standard codecs such as JPEG. Finally, the output images are tested to determine the effect of compression on object detection.

Before delving into the world of machine and deep learning, the concept of compression and its accompanying algorithms must be briefly introduced. Compression is defined as the process of encoding an image file in such a way that it consumes less space than the original image. Compression algorithms can be divided into two classes, lossless and lossy compression schemes. Lossless compression results in no loss of information such that the input and output images remain exactly the same. Lossy compression results in the loss of

some information, but yields a much higher compression ratio than is possible with lossless reconstruction [1]. In this paper, lossy compression is the main focus for its ability to yield high compression ratios.

There are many popular compression algorithms, but one of the most renowned is the JPEG codec. JPEG is a lossy compression algorithm that was the first international standard adopted for compression [2]. The JPEG standard is popular for the compression of lower bit-depth, 8-bit images. As imaging technologies have evolved and require more than 8-bits per color channel, the JPEG standard became insufficient for larger bit rates [3]. This resulted in a variety of lossless compression algorithms based upon JPEG to be created. As stated in [4], it is only natural in the course of progressive decoding to provide lossless compression. Thus, codecs such as JPEG2000 were developed that are lossless and allow for the compression of higher bit-depths [3].

The exponential growth of image sizes has resulted in the exploration of new compression techniques utilizing deep learning (DL). DL is a subsection of the larger field generally known as Artificial Intelligence (AI). The field of AI is generally interpreted as achieving computed biological intelligence. While this is the long term goal, the modern utilization of AI is far different. Figure 1.1 gives a general overview of the methods utilized in modern AI. Today, AI methods are utilized to improve a system in a given domain [5]. The applications are diverse, from generating weather models to natural language processing.

As noted in Figure 1.1 machine learning (ML) is also a subsection of AI, a level above DL. ML explores the study and creation of algorithms that can learn from input data and make predictions. ML is utilized in everyday life from spam-email filtering to traffic analysis. As the name implies, the deepest general class of AI is DL which utilizes layers

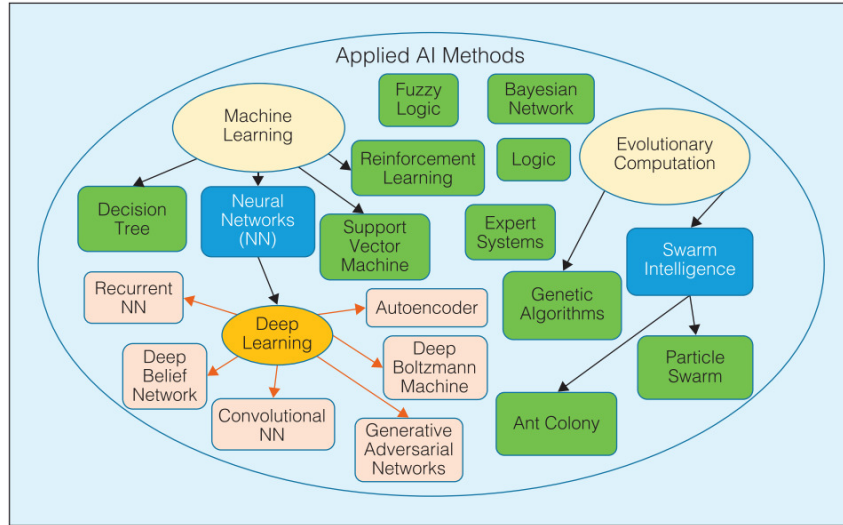


Figure 1.1: High Level Overview of Applied AI Methods

of neural networks and related machine learning algorithms, inspired by neuroscience [6]. The term DL refers to Artificial Neural Networks (ANNs) with multiple layers and are extremely popular as they can process vast amounts of data. DL wraps many different types of networks, from Auto-Encoders, General Adversarial Networks, to the focus of this paper Convolutional Neural Networks (CNNs).

CNNs get this name from the mathematical linear operation between matrices, convolution. The CNN has outstanding performance in machine learning problems, specifically applications that deal with large classification of data and computer vision [7]. A CNN at its core is a fully connected feed forward ANN or multi-layer perceptron. To obtain a solid grasp on CNNs, the three main layers are introduced. First, the convolutional layer slides different convolutional kernels along the input image. At each position, an operation sums the product of the kernel and projects that information onto a feature map. This reduces the size of the input image while also reducing the number of weights by removing irrelevant pixels [8]. The second, pooling layer reduces dimensionality by down sampling the data and

eliminating unwanted noise. Finally, the fully connected layer acts as a classifier, mapping the learned features into a sample markup space[7][8]. Figure 1.2 demonstrates the extrapolated reduced features from the input high-level feature demonstrated in classification.

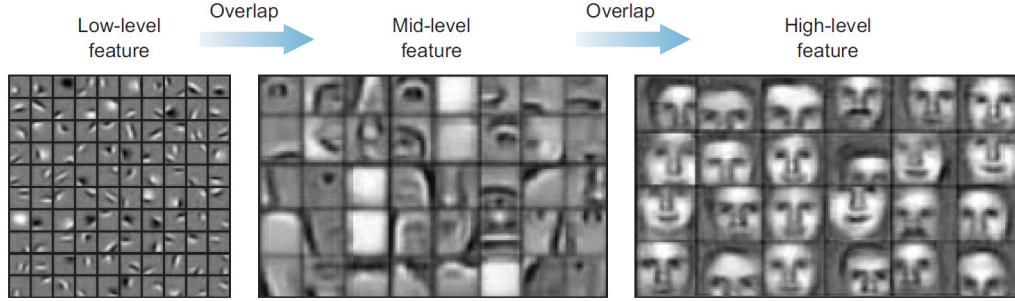


Figure 1.2: Low to High Level Learned Features from a CNN

Applications of CNNs are most notable in image detection and recognition. The classification features of the fully connected layer allow datasets to be categorized at a high level. The MNIST dataset [9] is a dataset containing hand-written digits. This utilizes CNNs to tackle applications such as optical character recognition, signature verification and interpretation of text. Another, the CIFAR-10 dataset [10] is an object detection dataset made up of 10 classes of natural images. Object detection algorithms are implemented in tandem with neural networks to classify and detect objects within the dataset. CNNs are developed and trained around these datasets as their class-oriented organization allows for ideal experimental conditions [11].

The CIFAR-10 and MNIST datasets utilize 8-bit RGB and 8-bit grayscale images respectively. Different image types and bit-depths can alter the performance and architecture of DL networks. As such, researching the affect of different data types such as 8-bit/14-

bit RGB, grayscale and infrared (IR) are important in validating the performance of the framework.

Nowadays, RGB is the most popular image type as most photographs are shot in color. However in the world of DL, applications such as pattern recognition require input images to be converted to grayscale. This is also the case with the CNNs in the thesis framework as they require a single channel input. Grayscale images are single channel monochrome images that's pixels can be typically represented by a single number. This number ranges from $[0,255]$ and identifies the intensity or luminance of an image. An RGB image has three layers, while pixels can be represented by three numbers sets (Red, Green and Blue) which are referred to as color spaces [12]. Due to the presence of fewer luminance bits, a converted grayscale image will have lesser contrast than the initial RGB image. It is important to utilize conversion methods that yield better visual perception to ensure good image processing performance. One such popular method is histogram equalization which spreads out the most frequent intensity values [13]. The conversion from RGB to grayscale can be seen below in figure 1.3. These images are captured on an electro-optical (EO) medium and also known as EO or visible light images

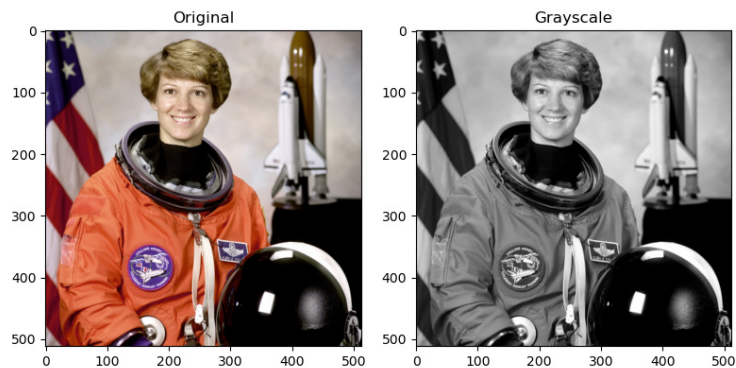


Figure 1.3: RGB to Grayscale Image Comparison

On the other hand, IR images are not captured in the visible light spectrum like RGB images. IR images can be represented as three channel or single channel images depending on the medium they are captured on. IR images are captured by an IR device which operates by an electromagnetic spectrum with a longer wavelength. The image is visualized by infrared radiation or heat given off by the object [14]. Figure 1.4 provides a side by side comparison of an IR and RGB image stitched together. It should be noted that the IR images explored in this thesis are of both 8-bit and higher bit-depth 14-bit images. The higher bit-depth is a result of the representation of a whole luminance range and color gamut [15]. Additionally, exploring these these higher quality images of larger bit-depth shed light on their compression performances and implementation challenges with DL models.

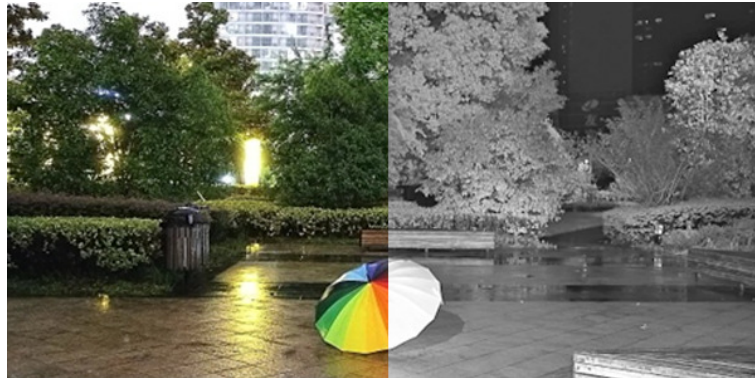


Figure 1.4: RGB to IR Image Comparison

IR, grayscale and RGB datasets are all utilized in training DL models for image processing applications, with each presenting its own unique set of challenges. CNNs must be optimized differently for each respective image type. Specifically the parameters that make up the networks must be altered to adapt accordingly. Parameter optimization of DL networks in itself is a popular research topic as different methods can vastly improve

performance. One such method by [16] used evolutionary algorithms to optimize hyperparameters within a neural network trained on the MNIST dataset. These algorithms behave biologically, "mutating" with different parameter combinations and removing the unfavorable mutations. Evolutionary algorithms have the ability to solve problems with non-linear, high-dimensional and multi-modal character. This makes them ideal for optimizing networks with a large number of unique hyperparameters. Another optimization method called Particle Swarm Optimization (PSO) behaves similarly to birds flocking together. PSO has extensive global optimization capabilities and is used to optimize the best number of hidden neurons, learning rate and initial weights [17].

The CONNECT framework consists of two CNNs for the compression and reconstruction of images. These networks have been deployed with specific parameters to ideally compress and reconstruct 8-bit grayscale images. In addition, image codecs also have parameters that can be altered to change the compression ratio and quality metrics of an image. Deploying an optimization algorithm on the framework increases efficiency in producing the best output images. This is the case for not only grayscale, but also experimenting with new data types such as IR which require different parameters. Optimizing the CNNs for different data types improves reconstructed output images and increase the versatility of model capabilities in compression.

The Computer Vision (CV) application of the compression framework is object detection (OD). Compression is a crucial facet in the realm of OD as detection strives to have live applications. Elon Musk's company Tesla is developing software for an autonomous vehicle powered by live OD technology. These live feeds require an enormous amount of data processing in which compression can help alleviate the load. In [18] a CNN is utilized to

directly target features of interest to be detected and compresses unnecessary features. This in turn reduces the computing cost to detect an object from the image.

1.1 Related Works

Compression in DL has garnered much interest helping provide a basis for the CONNECT framework and this thesis. The following related works shed context on recent developments in the field.

Both lossless and lossy compression has utilized DL algorithms to attain competitive performance. In [19] Toderici et al. approaches variable-rate lossy compression based on convolutional and deconvolutional LSTM networks. This framework consisted of a recurrent neural network (RNN) based encoder, decoder and neural network. From a more recent paper, Toderici et al. [20] proposed a compression network on images of arbitrary size. For a given rate of compression, the method would learn compression by reducing distortion in the images.

Theis et al. [21] proposed a lossy compression method that utilized compressive autoencoders. This was a unique approach as most methods utilize RNNs for compression as autoencoders typically were non-differentiable to compression loss. Bale et al. [22] utilized generalized divisive normalization for joint non-linearity utilizing additive uniform noise for better compression. Li et al. [23] proposed a compression method focused on the importance of the map of an image.

These methods all utilize deep learning to achieve strong performance but ignore the integration of image codecs which limits performance. The compatibility of both image codecs and deep learning models are the focus of the CONNECT framework.

CHAPTER II

DEEP LEARNING MODELS

This chapter provides a detailed insight into the framework and network that make up the project pipeline. Section 2.1 looks into the architecture details of the CONNECT compression framework and its origins. Section 2.2 provides a high level overview of the YOLOv5 [24] OD network and detection datasets. Finally, Section 2.3 sheds light on the overall project pipeline. This effectively shows how the compression framework and OD network are connected.

2.1 Compression Architecture Overview

The CONNECT framework is based upon Jian et al.’s paper “An End-to- End Compression Framework Based on Convolutional Neural Networks” [25]. This section details the model architecture designed by Jian et al. and subsequent results.

As detailed in figure 2.1, the framework consists of two CNNs and an image codec. The Compact Representation CNN (ComCNN) generates a compact representation of the image to be sent to the image codec encoder. This preserves structural information to allow for accurate reconstruction of high quality images. The Reconstruction CNN (RecCNN) is utilized to enhance quality of the decoded image and reconstruct the image as closely as possible to the input. The ComCNN and RecCNN share information with each other to achieve the highest image possible compression quality [25].

Sections 2.1.1 and 2.1.2 present the architecture and functionality of the ComCNN and RecCNN respectively. Section 2.1.3 provides an overview of the image codecs Jian et al.

tested. Finally, Section 2.1.4 details the functions utilized within the CNNs with Section 2.1.5 summarizing the Jian et al. framework results.

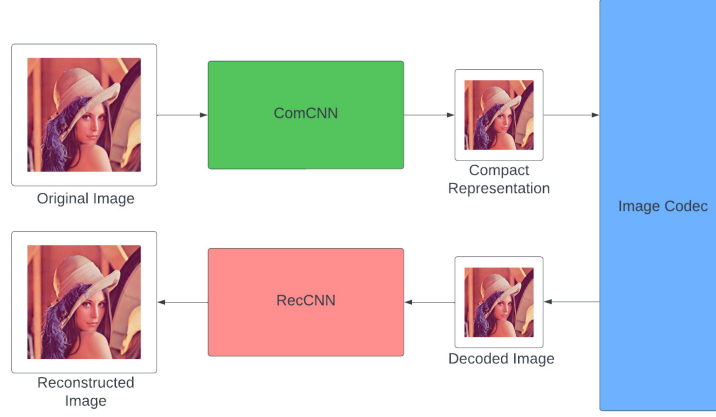


Figure 2.1: CONNECT Framework Image Processing Pipeline Overview

2.1.1 ComCNN

As seen in figure 2.2, the original image is fed into the ComCNN’s three weight layers. These weight layers are utilized to maintain the quality of the original image and output a structurally similar compact representation. The first layer of the ComCNN is used to perform patch extraction and representation. This extracts overlapping patches from the original image utilizing 64 feature maps, a kernel size of 3x3 to perform convolution and ReLU as the activation function. The second layer performs downscaling and enhances the features of the image. This is done with a stride size of 2 and the same filters as layer one. Finally the third layer utilizes a filter size of 3x3x64 to construct the compact representation. This process maintains the original image’s spatial structure to facilitate reconstruction of the decoded image using the RecCNN [25].

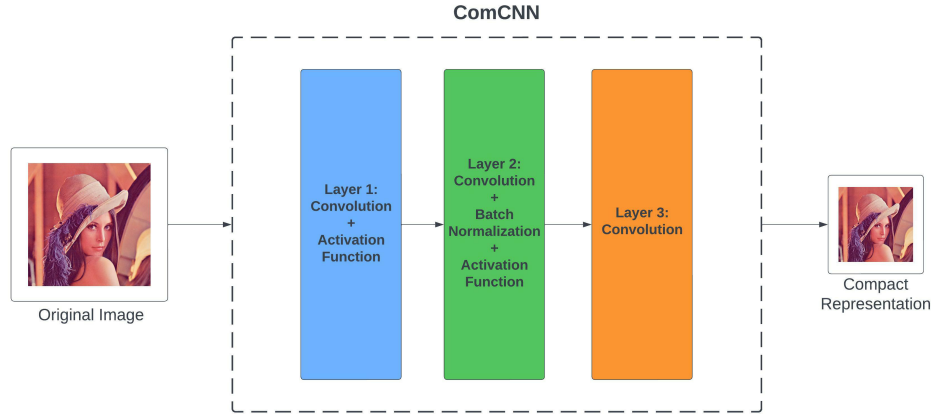


Figure 2.2: ComCNN Image Downscaling Pipeline

2.1.2 RecCNN

As seen in figure 2.3, the RecCNN is larger than the ComCNN with significantly more layers. It is composed of 20 weight layers to reconstruct the decoded image from the image codec. The decoded image is upscaled utilizing bicubic interpolation. It is then fed into the RecCNN which generates a residual image. The upscaled image and residual image are then summed to produce the final reconstructed image [25].

The first layer, similarly to the ComCNN, utilizes 64 filters of size 3x3 to generate 64 feature maps, followed by ReLU. The 2nd to 19th layers are utilized for feature extraction and use 64 filters of size 3x3x64, with batch normalization added between convolution and ReLU. Finally, the last layer is used to reconstruct the output with residual learning to speed up the process. This process outputs the residual image to be summed with the upscaled image [25].

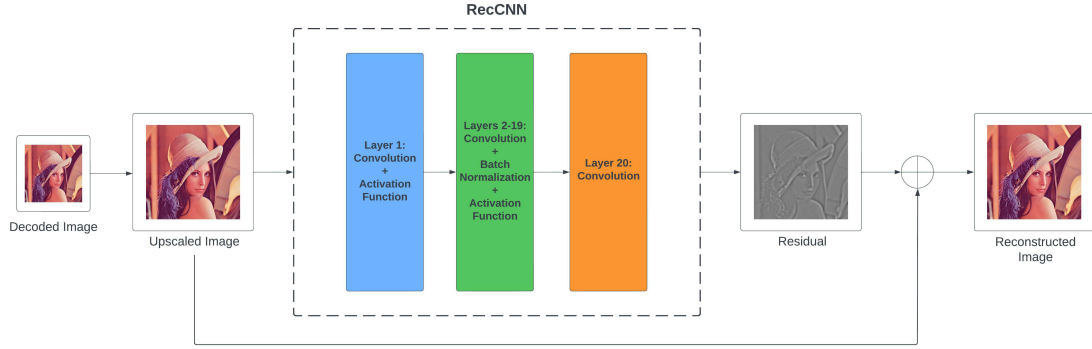


Figure 2.3: RecCNN Image Reconstruction Pipeline

2.1.3 Image Codecs

As previously mentioned, the framework is unique for its utilization and integration of an image codec. The image codec can be one of many, with Jian et al. testing primarily JPEG. The codecs can also have different outputs depending on the selected quality factor (QF). The QF controls the correlative compression and quality of an input image run through the codec [26].

The most popular codec utilized in the framework is JPEG [2]. JPEG is a lossy compression algorithm that offers state of the art compression for 8-bit images. The QF of JPEG ranges from 0 to 100, with a QF of 0 corresponding to high compression/low quality, and QF of 100 corresponding to low compression/high quality. Jian et al. tested the JPEG image codec at a QF of 5 and 10 against the framework [25].

Additionally, other codecs studied were JPEG2000 and PNG. JPEG2000 is a codec developed from JPEG as a lossless compression to tackle higher bit depths. This image codec compresses images up to 38-bits, a feature far ahead of standard JPEG [4]. The QF of JPEG2K ranges from 0 to 100 in a similar fashion as JPEG. Finally, the PNG image

codec uses the DEFLATE algorithm which supports lossless compression only. The PNG encoder complexity is very low but supports compression up to 16-bit images [3]. The QF range of PNG is 0-10 with 0 being the highest compression/lowest quality and 10 being the lowest compression/highest quality.

2.1.4 Compression Architecture Details

The following subsections provide an overview of the learning algorithm and loss function used by the framework and detailed in the Jian et al. paper. This provides a coherent understanding to the functionality of the framework without delving too deep into network parameters, which is discussed in chapter IV.

Learning Algorithm

The learning algorithm derived from the Jian et al. paper utilized by the framework is summarized in this subsection [25].

With the framework utilizing both the ComCNN and RecCNN to try and reconstruct the original image, the end-to-end optimization goal can be summarized by Equation 2.1.

$$(\theta_1, \theta_2) = \underset{(\theta_1, \theta_2)}{\operatorname{argmin}} ||Re(\theta_2, Co(Cr(\theta_1, x))) - x||_2 \quad (2.1)$$

In Equation 2.1, x represents the original image where θ_1 and θ_2 are parameters for the ComCNN and RecCNN respectively. Additionally, I represents the original image, $Cr()$ and $Re()$ represent the ComCNN and RecCNN respectively, with $Co()$ representing the codec. The function argmin is the point where the the function value is minimized and the double bars $||$ represent the Euclidean norm.

Replicating the summarized goal directly is not possible in practice as the image codec results in a rounding function that is non-differentiable. This complicates back-propagation making updating parameters more difficult as a result. Equation 2.2 uses the assumption that the RecCNN doesn't change with respect to the original input, which leads to updating the parameters as such.

$$\theta_1 = \operatorname{argmin}_{\theta_1} ||Re(\theta_2, Cr(\theta_1, I)) - I||^2 \quad (2.2)$$

Finally, Equation 2.3 was surmised to update the parameters of the RecCNN.

$$\theta_2 = \operatorname{argmin}_{\theta_2} ||Re(\theta_2, I_m) - I||^2 \quad (2.3)$$

The variable I_m represents the decoded compact representation of the original image. This is the image after it has passed through the ComCNN and the codec which in turn uses Equation 2.3 to update the RecCNN's parameters. These equations can be seen in more detail in [25].

Loss Function

The loss function of the framework is briefly detailed in this section.

Equation 2.4 derives the loss function of the ComCNN.

$$loss_{com} = \frac{1}{2}MSE(I_{rec}, I) \quad (2.4)$$

In Equation 2.4, I_{rec} represents the reconstructed image from the RecCNN and I represents the original image. The mean squared error is calculated between the two images.

Equation 2.5 derives the loss function of the RecCNN which is the more complex of the two.

$$loss_{rec} = \frac{1}{2}MSE(I_{res}, (I - I_{udcr})) \quad (2.5)$$

In Equation 2.5, I_{res} represents the residual image the RecCNN learns, I is the original image and I_{udcr} is the upscaled decoded compact representation. Similarly to the ComCNN, the mean squared error is calculated between the residual image and the original image, subtracted from the upscaled decoded compact representation. For a more detailed look at these equations, look to [25].

2.1.5 Compression Architecture Results

In the Jian et al. paper, the authors trained the framework for 50 epochs on 400, 180x180 sized grayscale images. To see the performance of the framework, seven 180x180 sized grayscale images were tested on. The image quality metrics PSNR and SSIM were observed to determine how successful the network is, with higher metric values indicative of better performance. Additionally, the networks were tested twice against a standalone JPEG test with a QF of 5 and 10. The QF of the JPEG codec within the framework was adjusted to a value that gave better results than the corresponding standalone JPEG QF test.

According to the Jian et al. results, the proposed framework outperformed JPEG in both SSIM and PSNR metrics. For a QF of 5, JPEG had a PSNR of 26.13 with a SSIM of 0.72 where the framework had a PSNR of 29.20 and SSIM of 0.83. For a QF of 10, JPEG had a PSNR of 28.99 with a SSIM of 0.81 where the framework had a PSNR of 31.40 and SSIM of 0.88. The framework outperformed JPEG for certain QF and was deemed a success in the utilization of CNNs in compression [25].

2.2 Object Detection Network Overview

OD tasks can be split up into the three categories of classification, detection and segmentation. Classification structures an image into a specific type of data, with a predefined category such as a string to serve as a description. Detection observes the image background and foreground to output a list of observed targets, usually represented by a rectangular bounding box. Finally, segmentation gives a pixel level description of an image to provide a low level understanding of each category. [27]

This section introduces the YOLOv5 object detection (OD) network, its features and testing results on the COCO dataset.

2.2.1 YOLOv5 Model Overview

The YOLOv5 OD network is a one-stage detector or a network that runs detections over a dense sampling of locations. This region-based detection network redefines OD as a regression problem making it one of the fastest open source detection models [27]. It's high processing speed has uses in real time detection from facial recognition to robot assisted apple harvesting.

The network is made up of three modules: the backbone, the head and the detection component. An overview of YOLOv5 and it's architecture can be seen in Figure 2.4 below. The backbone module of the network is a CNN that gathers and shapes image features at different levels of detail. Throughout the YOLOv5 network, center and scale prediction (CSP) Bottleneck formulates image features. The head implements PA-NET for feature aggregation on a series of layers to combine image features for the prediction process. Finally, the detection utilizes features from the head to take bounding box and class prediction

steps [28]. Figure 2.4 is a representation of the YOLOv5s network or small network. For the purposes of this thesis, the YOLOv5l network or large network is trained on which is a more dense version of the YOLOv5s network. The general architecture of both models is very similar with the YOLOv5l network having significantly more layers.

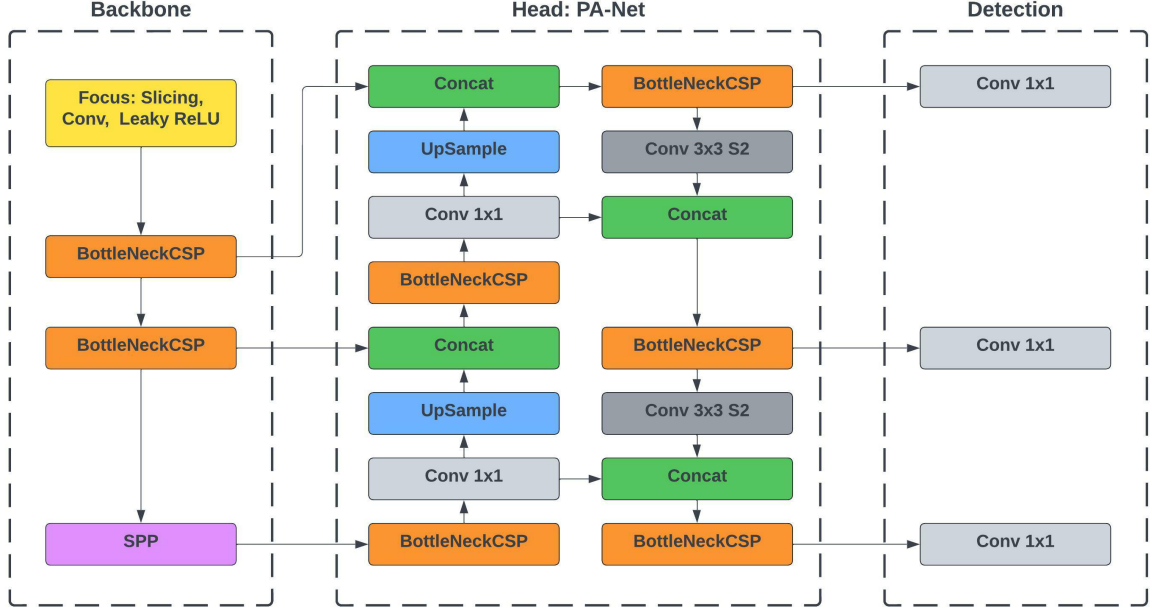


Figure 2.4: YOLOv5 Object Detection Network Pipeline Overview

2.2.2 Training and Datasets

The detection datasets OD networks are trained on are made up of images and annotations. The images are annotated and split up into training, validation and testing sets. The annotations typically contain bounding boxes and per-instance segmentation masks with various object categories. The YOLOv5 network only works with 8-bit images so locating a dataset with the proper bit-depth is imperative.

The Microsoft Common Objects in Context or COCO dataset is the most popular of the OD datasets, released by Microsoft in 2014. It contains 164k RGB 8-bit annotated images split into 82k training images, 41k validation images and 41k testing images [29]. Figure 2.5 provides a sample image with annotations from the COCO dataset for reference. This dataset is used as a baseline for best case detection results in this thesis.

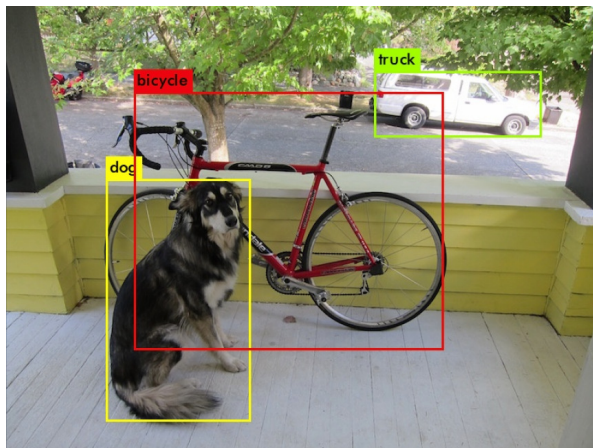


Figure 2.5: Example of Bounding Box Object Detections in the COCO dataset

The ADAS FLIR [30] dataset is studied in this thesis for both compression and OD purposes. The FLIR dataset provides 26,442 fully annotated IR images with 15 classes for the development of OD networks. The dataset is also split 50/50 between 8-bit and 14-bit IR images. Additionally, the training/testing split for each bit depth is a 90/10 split. This is ideal for low and high bit depth compression with the added benefit of low bit depth object detection.

2.2.3 YOLOv5 Model Results

For best case results, Abel et al.'s paper [31] testing the detection network on COCO will be set as a baseline. The paper deployed the YOLOv5 model trained on pantry image scenes from the COCO dataset. The results yielded a mAP(0.5) score of 0.9948 at 87 epochs of training and a detection accuracy of 100 percent. These results show the state of the art performance of the YOLOv5 network.

2.3 General Architecture Overview

This brief section elaborates on the specific goals of the framework and detection network respectively. Figure 2.6 provides a look at the general pipeline between the CONNECT framework and YOLOv5 network. It should be noted that the compression framework is completely independent of the OD network. The optimization of hyperparameters will occur solely within the CONNECT compression framework. The YOLOv5 model is simply an application and or tool to be utilized for detection purposes. Following the full optimization of the CONNECT framework, the ideally compressed images will be tested against baselines to determine how effective DL compression is in the application of OD.

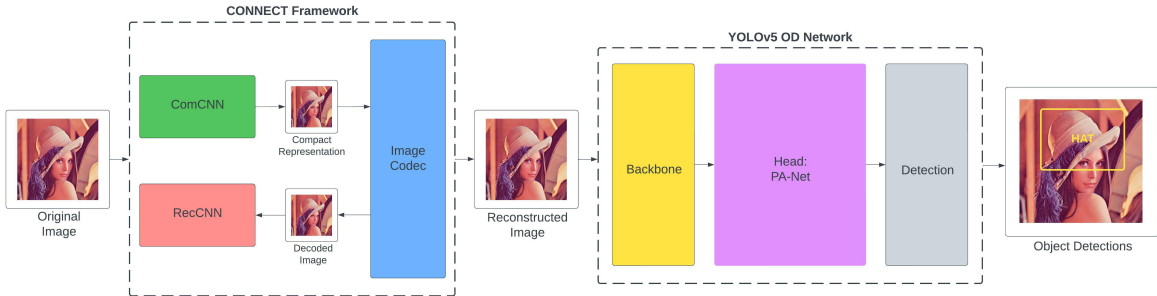


Figure 2.6: CONNECT Framework to YOLOv5 Network General Architecture Pipeline

CHAPTER III

MODEL PERFORMANCE EVALUATION

This chapter provides different analytical methods in which to determine the success of each model. There are a variety of metrics that can be studied to identify the desired results. The focus of CONNECT framework is to yield a high compression value while maintaining competitive image quality. This requires the image to have not only solid quality and compression metrics, but also a comparable visual perception. Additionally, the success of these metrics are imperative for ideal performance in the YOLOv5 network. OD itself has a series of detection metrics to be evaluated in addition to image quality and compression metrics. These quality standards are discussed in detail and evaluated by their importance in accurately portraying the results of each model.

3.1 Image Quality Metrics

There are a variety of quality metrics to observe the different characteristics of an image. In addition, the human perception of images has to be taken into account when determining which metrics are most effective. This divide can be split up into two different categories of perception, subjective and objective. The image quality metrics in this section highlight the best objective methods based on explicit numerical criteria for determining the results of a processed image [32].

The three image quality metrics in this section have been deemed to carry the most objective weight in the subjective perception of an image. In other words, the quality metrics are appropriate for matching the human perception of an image. The metrics evaluated are the mean square error (MSE), peak signal to noise ratio (PSNR) and structural similarity index (SSIM).

3.1.1 Mean Square Error

The goal of MSE is to measure the signal fidelity or provide a quantitative score that identifies the level of difference or fidelity between two images. MSE is the dominant quantitative metric in the field of image processing for several reasons. It is simple, parameter free and computationally inexpensive making it perfect for large scale applications. Additionally, it has a clear physical meaning as the natural way to determine the energy of the error signal [33]. The equation for MSE can be seen below in Equation 3.1.

$$MSE(x, y) = \frac{1}{N} \sum_{i=1}^N (x_i - y_i)^2 \quad (3.1)$$

The variables x and y can be described as two finite-length discrete signals. N is the number of signal samples, or pixels in the case of images, where x_i and y_i are the values of the i th sample in x and y respectively. The best possible value that MSE can output is zero, with a lower value implying a smaller error [34]. MSE is a widely used metric but can be problematic in cases where it does not always match human perception.

3.1.2 Peak Signal to Noise Ratio

PSNR is another widely used image quality metric, utilizing MSE for its calculation. The PSNR value approaches infinity as MSE approaches zero, so the higher the PSNR value the higher the image quality [33]. The formula for calculating PSNR can be seen below in Equation 3.2.

$$PSNR(x, y) = 10 \log_{10} \left(\frac{L^2}{MSE(x, y)} \right) \quad (3.2)$$

Similar to MSE the variables x and y are representative of two finite-length discrete signals, or the original and test image respectively. L is representative of the dynamic range of pixel intensities. If based on 8-bit image, L would have a value of $L = 2^8 - 1 = 255$, which can be re-written as Equation 3.3 below [32].

$$PSNR(x, y) = 10 \log_{10} \left(\frac{255^2}{MSE(x, y)} \right) \quad (3.3)$$

For the purposes of this thesis, PSNR is heavily utilized for an objective analysis of both 8-bit and 16-bit images. Since PSNR has a similar shortcoming of human perception as MSE an additional metric, SSIM, is utilized in tandem.

3.1.3 Structural Similarity Index

The final image quality metric utilized is SSIM which calculates the quality of imperceptibility in stenographic images. According to [32], PSNR and SSIM are analytically linked while providing different objective perspectives on the quality of an image. SSIM was built upon the three main factors of luminance, contrast and structure. These factors replace the summing method used as the basis for deriving PSNR [35]. The formula for calculating SSIM can be seen below in Equation 3.4.

$$SSIM(x, y) = [l(x, y)]^\alpha \cdot [c(x, y)]^\beta \cdot [s(x, y)]^\gamma \quad (3.4)$$

Similarly to MSE and PSNR the variables x and y are representative of two finite-length discrete signals, or the original and test image respectively. The luminance function or $l(x, y)$ is estimated as the mean intensity. The contrast function or $c(x, y)$ is calculated by

removing the mean intensity from the signal and computing the standard deviation. Finally, the structure function or $s(x,y)$ is calculated by dividing both signals by their standard deviation. The exponents of α , β and γ correlate to the weights or relative importance given to each function. The SSIM value can range from $[-1,1]$ with 1 being the best SSIM value [35].

For the purposes of this thesis, the PSNR and SSIM are both taken into account when objectively analyzing processed image results.

3.2 Image Compression Metrics

Image compression metrics are much simpler to calculate than quality metrics as they are derived from image size and pixel ratios. Nevertheless, these metrics are extremely important in determining the success of the CONNECT compression framework. They allow for an objective analysis into the success of the compression framework and share equal importance to the image quality metrics.

3.2.1 Bits Per Pixel

The amount of information in an image can be calculated by the number of bits per pixel (BPP). This is important in calculating the overall size and memory cost to store an image. BPP is calculated by taking the ratio of bits in the compressed image (C_{bits}) in comparison to the total number of pixels in the original image (O_{pix}) [34]. This formula can be seen below in Equation 3.5.

$$BPP = \frac{C_{bits}}{O_{pix}} \quad (3.5)$$

3.2.2 Compression Ratio

Similar to BPP the compression ratio or CR is the ratio of the size of the uncompressed image (I_u) to the size of the compressed image (I_c). A higher compression ratio lessens the storage cost of an image by reducing its size [34]. The formula for this can be seen below in Equation 3.6.

$$CR = \frac{size(I_u)}{size(I_c)} \quad (3.6)$$

The CR metric is utilized heavily throughout this thesis as the main indicator of successful compression. It's simplicity allows for experimental repeatability and is the ideal compression metric.

3.3 Object Detection and Classification Metrics

The following section provides an overview to detection and classification metrics that determine the success of the YOLOv5 OD network. Additionally, some background concepts are discussed to shed context on the metrics. The metric most frequently used for evaluating model performance is accuracy, however is misleading and highly prone to unbalanced data in OD applications. The metrics in this section are proven to be ideal for OD predictions and determining model success. The OD metrics utilized are precision, recall, F-1 Score and mAP score.

3.3.1 Confusion Matrix

The confusion matrix is the basis of classification metrics as most formulas are derived from its results. A confusion matrix plots resulting detections and compares the actual

results against the predicted results. The matrix consists of four metrics which can be seen in Figure 3.1 below. A true positive (TP) is a hit on the target as it correctly indicates the

		Predicted Condition	
Total Population = P + N		Positive (PP)	Negative (PN)
Actual Condition	Positive (P)	True Positive (TP), hit	False Negative (FN) type II error, miss, underestimation
	Negative (N)	False Positive (FP), type I error, false alarm, overestimation	True Negative (TN), correct rejection

Figure 3.1: Confusion Matrix Calculations for Classification Metrics

presence of a detection. On the contrary a true negative (TN) is a correct rejection as it correctly indicates the absence of a detection. A false positive (FP) incorrectly indicates the presence of a detection and is considered a type I error and/or an overestimation. A false negative (FN) incorrectly indicates the absence of a detection and is considered a type II error and/or underestimation. [36]

3.3.2 Precision, Recall and F-1 Score

The object classification metrics of precision, recall and F-1 score are all used for classification problems. These metrics are adaptations of the classification capabilities of the confusion matrix. The following section highlights the formulations and benefits of each metric.

Precision reports on the exactness of a model, as it calculates the number of correct results over all the returned results. A high precision is indicative of a high positive predicted rate with a low number of FP [37]. Precision is formulated by dividing the amount of TP by the sum of the TP and FP as seen in Equation 3.7.

$$Precision = \frac{TP}{TP + FP} \quad (3.7)$$

Recall reports on the completeness of a model by computing the number of correct results over the number of results that should have been returned. The metric is best when capturing the majority of the minority instances with FN [37]. Recall is formulated by dividing the amount of TP by the sum of the TP and FN as seen in Equation 3.8.

$$Recall = \frac{TP}{TP + FN} \quad (3.8)$$

F-1 Score reports on the moderateness of a model, representing the harmonic mean between precision and recall. F-1 Score is the weighted average of precision and recall, best utilized where a moderation of FP and FN is required [37]. The weighted average calculation can be seen in Equation 3.9.

$$F = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (3.9)$$

3.3.3 mAP Score

Object Detection adds another layer of difficulty for classification and detection metrics. Bounding boxes are incorporated to identify a target along with proper classification. This results in multiple objects and classifications within a single image or frame [36]. The mAP

score metric is the best detection metric for this task and is calculated in three steps. First the Intersection Over Union (IOU) threshold must be determined.

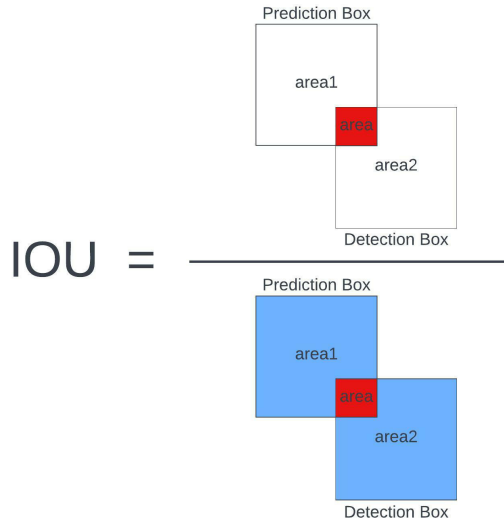


Figure 3.2: Visualization of the Intersection Over Union Calculation

Figure 3.2 shows IOU is calculated by dividing the intersected area between the bounding boxes by the area in union of the bounding boxes. Next, the average precision AP is calculated by taking the area under the precision-recall (PR) curve seen in Equation 3.10 below, where n is the number of thresholds [36].

$$AP = \sum_{k=0}^{k=n-1} [Recall(k) - Recall(k+1)] * Precision(k) \quad (3.10)$$

Finally, the mAP score is calculated by computing the mean of the AP where n is number of classes and AP_k is the AP class of k seen in Equation 3.11

$$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_k \quad (3.11)$$

The mAP score is the detection metric studied for the purposes of this thesis as it provides an overview to all previously discussed metrics. It takes into account not only detection but classification metrics to provide a well rounded analysis on model performance.

3.4 Visual Perception

The final evaluation criteria for model performance is visual perception. This task is carried out for the CONNECT compression framework as it provides image outputs for this exact purpose. As seen in Figure 3.3, the inputs and outputs throughout the framework are saved for subjective analysis. The images are as follows: the input image is sent into

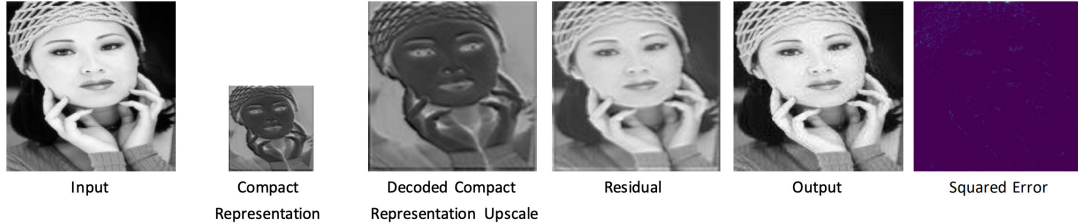


Figure 3.3: CONNECT Framework Image Outputs for Visual Perception

the ComCNN which outputs the compact representation. This is then fed into the image codec which outputs the decoded compact representation to be upscaled. The upscaled image is fed into the RecCNN which outputs the residual to be summed with the former. This sum yields the output image, with a MSE being calculated between the input and the

output. This process is observed subjectively with visual perception and objectively with image quality and compression metrics.

CHAPTER IV

HYPERPARAMETER EXPLORATION

The following chapter focuses on the optimization of hyperparameters within the CONNECT compression framework using Optuna. The optimization of CNNs is a popular topic in academia, as replacing the monotonous process of selecting hyperparameters is extremely beneficial. Multiple approaches have been devised, such as [38] and [16] using differential evolution with parallel subpopulations to find an efficient architecture for CNNs. Particle swarm optimization was studied by [17] for its extensive global optimization capabilities and use to optimize the best number of hidden neurons, learning rate and initial weights. For this thesis, the Optuna framework is utilized to wrap the CONNECT framework and optimize its corresponding CNNs. Optuna was selected for its relative newness and ease of integration with the Python heavy framework. It's multi-modal means of optimization and overall functionality are detailed in this chapter.

4.1 Optuna Framework

The Optuna Framework is defined by three base criteria. The first is the framework uses define-by-run programming that allows a user to dynamically create a search space. The second is the implementation of an efficient sampling and pruning algorithm that allows for user-customization. The third criteria is the framework is easy to setup and can be deployed to tasks of various types [39].

Optuna is the first of it's kind in utilizing a define-by-run approach. The method allows the user to define any combination of hyperparameters for a set test or run. This versatility was one of the main factors for the integration of the framework into CONNECT. At it's simplest, Optuna receives a set of hyperparameters to test. Each test of different

hyperparameter combinations is called a trial, which outputs an objective value. This objective value is the value to be optimized towards and Optuna will adjust hyperparameters accordingly to reach the best value [39][40]. This basic overview can be seen in Figure 4.1 and is further explored in this section.

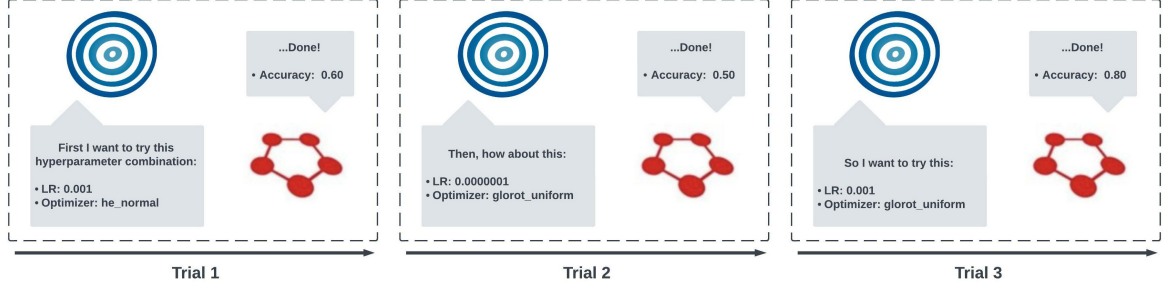


Figure 4.1: High Level Optuna Optimization Process Flowchart

The Optuna framework can be broken down into three main steps, the objective function, hyperparameter selection and the creation of an Optuna study.

The objective function contains the machine learning logic of the program, which in the case of the thesis is the ComCNN and RecCNN. The objective function returns an evaluation score known as an objective value or the metric being optimized. In this paper, the objective function containing the CNNs optimizes towards the objective value of PSNR, CR and SSIM. The two types of objective functions that can be called are single-objective and multi-objective functions. A single-objective function maximizes or minimizes a single objective value. Single-objective functions are the main feature of Optuna as they allow for additional algorithms and plot features to be utilized. The optimization algorithm used by a single-objective function in this paper is a Tree-structured Parzen Estimator (TPE) sampler. A TPE sampler is a sequential model based optimizer that approximates

hyperparameter selection based on historical performance [40]. A multi-objective study is utilized to maximize towards several objective values at the same time. The optimization algorithm used in this thesis for multi-objective studies is a grid sampler, which allows for the specific selection of hyperparameter combinations.

Hyperparameter selection is straightforward and can be chosen from a variety of data-types. These include but are not limited to string, float, integer and boolean datatypes. When the variables are defined as hyperparameters to be optimized, a range of values must be selected within each variable. This range will be utilized to select combinations of each hyperparameter and can be a numerical range or a list of strings. In single-objective studies, it is possible to plot a chart that defines the importance of each hyperparameter. This is crucial in determining which parameters hold the most weight in the optimization process [39][40].

Finally upon completion of the objective function and hyperparameter selection, an Optuna study is created. The Optuna study is the function that cycles through the hyperparameter combinations one trial at a time. The study is user customized with several features starting with the number of trials or times the hyperparameters should be tested. Additionally, a sampler or optimization algorithm is selected, which in single-objective studies is a TPE sampler, and in multi-objective studies is a grid sampler. The direction of the study can be changed to maximize or minimize the evaluation score or objective value. Finally a pruning function can be called in single-objective studies to skip over trials that do not reach a certain objective value criteria. This is imperative for saving time in the case of a study with a larger number of hyperparameter combinations [39][40].

4.2 Variance and Non-determinism

The repetitive nature of Optuna begs the question whether hyperparameter combinations are consistently reproducible. This is not always the case due to the prevalence of variance, caused by non-determinism in DL models trained on GPUs. DL models use non-determinism to improve model training time and accuracy [41]. Hence with non-determinism, multiple identical training runs with the same data and network will produce different model accuracies and training times. In addition to this, DL libraries such as PyTorch and TensorFlow introduce additional variance due to floating-point computations and parallelism [42].

To reduce the prevalence of variance in DL models, two approaches are utilized. The first is the law of averages, which is used in studies with a small number of hyperparameter combinations. Multiple studies with the same hyperparameters will be tested and their objective value averaged out. The second approach is to conduct a series of smaller studies involving less complex training data to establish a baseline. A second larger study will then be conducted with a smaller number of hyperparameters to reduce complexity.

4.3 Model Hyperparameters

This section details the hyperparameters chosen to be optimized in the CONNECT network and the baseline values set by Jian et al [25] for CONNECT. Additionally, it is important to note the function of each hyperparameter within the network to determine its impact. This assists with selecting the proper range of values or categories to be studied. The reason for the selection of each hyperparameter and their importance within the CONNECT compression framework is discussed.

4.3.1 Learning Rate

The learning rate (lr) is the hyperparameter that controls how much the weights of each network are adjusted. This can be a single unchanged lr value as the model iterates through training epochs or a scheduled lr that changes [43]. The baseline lr hyperparameter in CONNECT is split into two distinct hyperparameters for each CNN. The ComCNN uses a decaying lr from 0.01 to 0.0001 where the RecCNN lr is decayed from 0.1 to 0.0001 [25]. Both lr will be optimized within their respective ranges to determine if the ComCNN and RecCNN weights are updated in the most effective way, with regards to their network architectures.

4.3.2 Number of Layers

The number of layers within a CNN determines how deep the network is. Deeper neural networks have more neurons and connections between layers that can alter model performance [44]. The RecCNN has 18 identical feature extraction layers selected by the Jian et al. team and utilized in the CONNECT framework. It should be noted the Jian et al. team tested deeper networks at only negligible performance improvements [25]. The RecCNN's number of layers will be studied as a hyperparameter, with a range of 14-24 layers to determine if 18 layers is ideal for feature extraction.

4.3.3 Kernel Size and Strides

The kernel size is the filter size upon which a convolution looks at a data matrix or pixels in the case of an image. A kernel size of 3 or 3x3 indicates the observation of an area of 9 pixels. Strides are the number of pixels by which the kernel size slides per convolutional observation [45]. The kernel size selected by Jian et al. and CONNECT for

both the ComCNN and RecCNN are 3x3. The stride size for the ComCNN is set to 2 for downscaling while the RecCNN is set to 1 [25]. A kernel size of 1, 3, 5 and 7 will be experimented with as a hyperparameter for both the ComCNN and RecCNN. Additionally, the stride sizes of 1 and 2 will be tested in different hyperparameter combinations to confirm whether they're ideal.

4.3.4 Initializer

An initializer is an equation that sets the network weights to a small random number. This allows for a random distribution of weights eliminating any starting biases. Xavier has been proven to be a successful initializer in a variety of DL models [46]. The Jian et al. paper utilizes the He_normal initializer function [25]. For the hyperparameter search in this thesis, both He_normal and the Glorot_uniform or Xavier initializer will be studied.

4.3.5 Activation Function

The activation function is a formula that calculates the weighted sum of a neuron and determines if it should be activated or not. This introduces non-linearity into the neuron and works in tandem with updating its respective weight and bias. [47] [45]. The Jian et al. paper selected the ReLU activation function for its versatility and popularity [25]. The activation functions selected for hyperparameter optimization were based on their general popularity in academia. The activation functions to be studied are ReLU, Elu, Softmax and Tanh.

4.3.6 Quality Factor

The final hyperparameter to be studied is not a DL parameter but that of the JPEG and JPEG2000 image codecs. The QF of the image codec determines the quality and compression of a decoded image [26]. The Jian team selected the QF to be whatever framework output was comparable to a JPEG QF of 5 and 10. For hyperparameter exploration, all JPEG and JPEG2000 QF will be explored from 0-100 in steps of 5. This provides an insight into the performance of the CONNECT model against base JPEG. Additionally, this hyperparameter will be explored following the optimization of all DL hyperparameters.

CHAPTER V

EXPERIMENTAL SETUP AND RESULTS

This chapter details the experimental setup used for model training and testing, along with subsequent results. A description of the experimental setup provides context on the medium which model evaluations are carried out. This setup consists of the software, operating system (OS) and hardware utilized for testing. Additionally, a high level overview of the training and testing notebook flowchart is provided. The results yielded from the experimental setup are divided into three categories, general optimization, QF analysis of optimized models and object detection analysis. These results are further divided into the datatype of the images experimented on.

5.1 Experimental Setup

This section evaluates the experimental setup used to develop the CONNECT framework and run the YOLOv5 OD network. The Ubuntu OS and Jupyter Lab provide the platform on which the software is developed. Each Jupyter notebook is written in Python and trained on dedicated GPUs. A flowchart is provided and discussed in detail with regards to the specifics of the DL training and testing process. Additionally, the datasets utilized are listed along with their corresponding training/testing split.

5.1.1 OS and Software

Ubuntu [48] is Linux distribution based on Debian, primarily made up of open source software. This is the ideal environment to carry out research due to its flexibility as an OS. For software development, an IDE known as Jupyter Lab [49] is used for its web-based interactive software development capabilities. A notebook interface is used for literate

programming, written in the Python [50] development language. Python is chosen for its wide variety of DL libraries available and ease of use as a programming language.

The DL libraries used to develop the compression framework and OD network are TensorFlow [51] and PyTorch [52] respectively. TensorFlow is an open source library used for machine learning, with a particular focus on training deep neural networks. PyTorch is also an open source library for machine learning, but has a greater focus in developing computer vision and natural language processing applications. Hence, the CONNECT CNNs are fully developed on TensorFlow and the YOLOv5 network [24] on PyTorch.

Finally, for the optimization of hyperparameters, the Optuna [40] framework is used to wrap the CONNECT compression framework. Optuna can identify variables used within TensorFlow and find the ideal combination. This process poses a challenge due to a high volume of results, which are subsequently stored in a SQL server for efficiency.

5.1.2 Hardware

Training a DL model is a computationally expensive process and requires an advanced hardware setup to be efficient. For this project, the DL models are trained on two high performance machines known as Chambers and Roland. Chambers consists of two Intel(R) Xeon(R) CPU E5-2620v4 @ 2.10GHz and 128GB of DDR4 RAM. The bulk of training is performed on its four NVIDIA Titan XP GPUs with 3840 CUDA cores. Roland consists of an Intel(R) Core(TM) i9-10980XE CPU @ 3.00GHz with 256 GB DDR4 RAM. Training is performed on its two NVIDIA RTX 3090Ti GPUs with 10496 CUDA Cores. The majority of training is performed on Roland for its superior 3090Ti GPU, which at best is twice as fast as the Titan GPU.

Training time varies depending on the depth of the model and batch size of the selected dataset. In this thesis, small dataset batch sizes are used to reduce training time for an initial perspective on model performance. Larger batch sizes with deeper models can take up to 20 hours to train. The training time of each Optuna run is provided for a perspective on the depth of DL models.

5.1.3 Training and Testing Pipeline

The pipeline for integrating Optuna into the CONNECT framework was developed over the course of several months. Alongside the optimization of hyperparameters, the process by which to achieve results was also made as efficient as possible. The training and testing pipeline has four main goals. The first goal is to train the CONNECT framework and achieve the most optimized model with Optuna. The second goal is to test the optimized CONNECT model for quality metrics, compression metrics and visual perception. The third goal is to test different QF values of CONNECT against base JPEG. Finally, the fourth goal is to test OD results of CONNECT against base JPEG, alongside additional QF analysis.

To achieve the proper results, a pipeline is developed with different training and testing notebooks. An input image is sent to the CONNECT training notebook, where the CNNs are optimized for that datatype. The input image is also sent to a JPEG testing notebook where quality and compression metrics are gathered for a range of QF values. The optimized CONNECT model is also tested in a separate testing notebook where quality and compression metrics are gathered for a range of QF values. Additionally, this testing notebook provides a visual perception output. It should be noted each optimized model is saved and utilized for QF analysis, as to not retrain the model fully for every QF tested.

This utilization of transfer learning cut down training time by almost 80 percent. Finally, the output images from the JPEG testing notebook and Optimized CONNECT model are sent to the YOLOv5 OD network. These images are run through the network and compared side by side with quality, compression and detection metrics for different QF values. Figure 5.1 is a flowchart that details this process to achieve proper results.

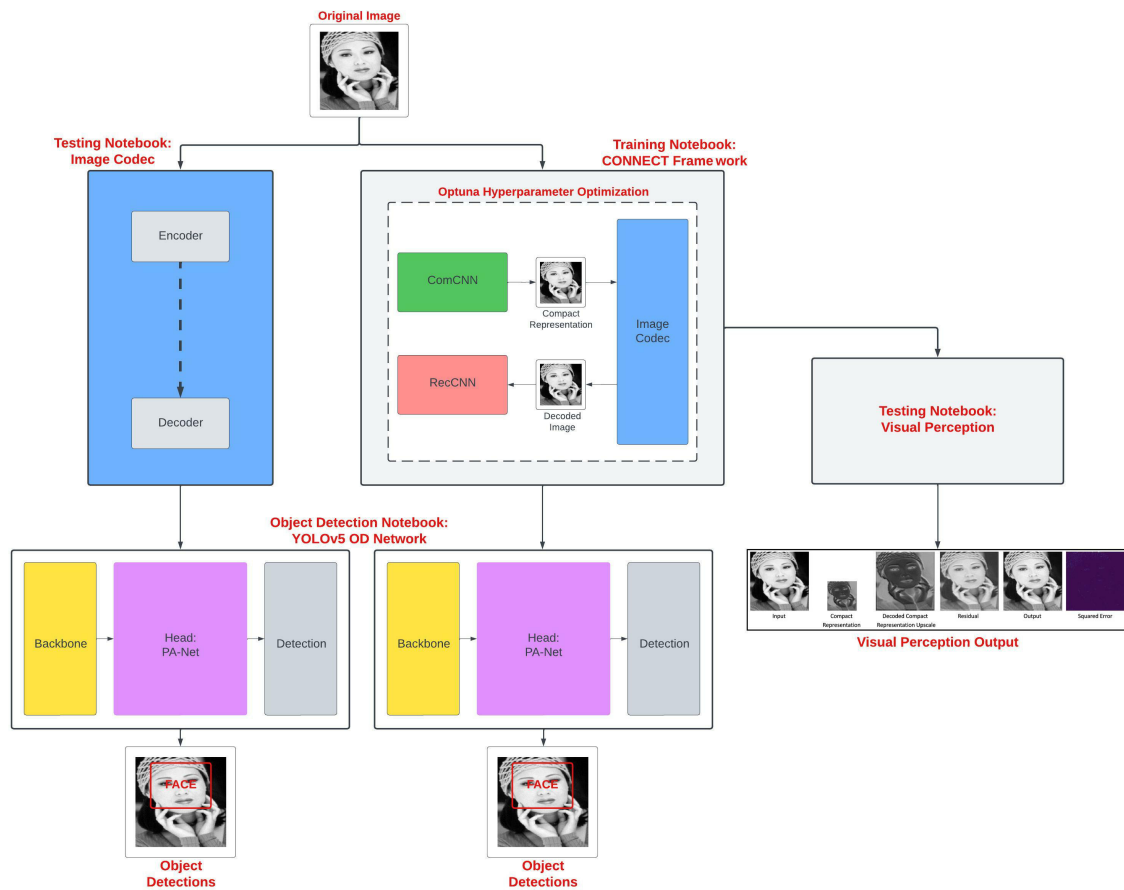


Figure 5.1: Optuna Training and YOLOv5 Network Testing Flowchart

The images used for training and testing come from four popular datasets, all of which are converted to grayscale.

For EO images, BSD400 [53], a terrestrial dataset of 400 RGB 481 x 321 size images, is used for CONNECT framework training. The dataset is split into 80 percent training and 20 percent validation sets respectively. Set5 [54], a dataset of five RGB 512x512 size images, is used for CONNECT framework testing. For OD in the EO domain COCO128 [29], a dataset of 128 RGB 640x480 size images with 80 classes, is used to train the YOLOv5 network. The dataset is then applied to the compression framework and the output is tested on the OD network.

For IR images, ADAS FLIR [30], a dataset of 26,442 fully annotated IR images of size 640 x 512 with 15 classes is used for training, testing and OD. The dataset contains two sets of 9,711 training and 1,312 validation images of 8-bit and 14-bit depths. These datasets are used for training the CONNECT framework while a smaller subset of the dataset is used for the testing notebook. Since the images are all annotated, they're also used to train the YOLOv5 network. Similar to EO, the dataset is applied to CONNECT and the output is tested on the YOLOv5 OD network.

5.2 General Model Optimization

This section provides the results of an in depth hyperparameter search to optimize CNNs of the CONNECT framework. The following figures and charts consist of hyperparameter combinations, also called models for simplicity, that yield the best quality and compression metrics. Typically, there are hundreds of hyperparameter trials conducted, with the chosen models performing at a similar or better level than the baseline Jian hyperparameters. These models are analyzed with image compression and quality metrics alongside visual perception to determine which models are best for their respective datatype. Hyperparameter importance charts are included with the initial single-objective studies in 8-bit EO and

14-bit IR. This feature is unnecessary for the multi-objective studies of the 8-bit IR domain as the most important hyperparameters have already been identified.

5.2.1 8-bit EO

8-bit EO was the first datatype introduced to the Optuna hyperparameter optimization process. Through trial and error, the Optuna CONNECT to YOLOv5 pipeline was created through groundwork lay by 8-bit EO experimentation.

The method to optimize CONNECT hyperparameters in the 8-bit EO domain is a slower process than 14-bit and 8-bit IR. The overall list of hyperparameters to be optimized was gathered over several months hence experimentation is performed one group at a time. The 8-bit EO studies are single-objective, optimized towards the average PSNR value of the output images. The Optuna pruning feature is also incorporated to skip any under-performing trials. 8-bit EO studies are all trained on the full batch size BSD400 dataset for 50 epochs.

Before utilizing Optuna for optimization, the Jian et al. [39] hyperparameters are trained at 50 epochs to establish a baseline. The base hyperparameters and output metrics for the Jian model can be seen below in Table 5.1. This establishes baseline image quality and compression metrics with a PSNR of 29.68, CR of 2.33 and SSIM of 0.8493. The initial criteria is to select the best performing model in terms of PSNR, CR and SSIM.

The first group of parameters trained in Optuna are the decaying learning rates (lr) of the ComCNN and RecCNN. A range of 0.1 to 0.0001 in steps of 0.01 is implemented as the range to be explored. The lr model study is trained for 50 trials, which at 50 epochs on two Roland GPUs takes 75 hours with total of 10 trials pruned. The resulting best trial

Model	Learning Rate (Com,Rec)	Activation Function	Initializer	Number of Middle Layers	Stride Com/Rec	Kernel Size	PSNR	CR	SSIM
LR Model	(0.006,0.0006) (0.002,0.0007)	ReLU	He_normal	18	2/1	3	26.36	2.21	0.7984
Act/Int Model	(0.01,0.0001) (0.1,0.0001)	ReLU	Xavier	18	2/1	3	29.69	2.34	0.8487
Lay/Str/Ker Model	(0.01,0.0001) (0.1,0.0001)	ReLU	Xavier	16	1/1	1	40.69	1.58	0.9873
Jian Model	(0.01,0.0001) (0.1,0.0001)	ReLU	He_normal	18	2/1	3	29.68	2.33	0.8493

Table 5.1: 8-Bit EO Model Optimization Training Results

is a ComCNN lr of (0.006, 0.0006) and RecCNN lr of (0.002, 0.0007) which yields a low PSNR of 26.36. Additionally the CR and SSIM under-performed the baseline, so the lr hyperparameters are frozen with initial Jian values.

The second group of parameters trained in Optuna are activation functions and initializers. Since only a total of eight combinations are possible, a total of 24 trials are performed to have a minimum of three trials per combination. The average of these combinations is taken in an attempt to reduce inconsistencies from variance in the models. At 50 epochs for 24 trials on two Roland GPUs, the study takes 72 hours total. The resulting best averaged model is the base ReLU activation function and Xavier initializer. The model slightly outperforms the Jian model parameters in PSNR and CR while under-performing in SSIM. Since the results are slightly better, ReLU and Xavier are frozen to be utilized in the next model. However, it should be noted that the difference in results between this model study and the Jian model is almost negligible.

The third and final group of parameters trained in Optuna are the number of middle layers, strides and kernel size. During this study, the hyperparameter importance chart is plotted to show the parameters that had the biggest impact on the model, seen in Figure

5.2. The importance is determined from the fANOVA algorithm, which fits a random forest regression model that predicts the objective value given a parameter configuration [40]. The initial lr are determined to be the most important to the success of the model, validating the lr parameters set in place by Jian et al. A smaller range of middle layers

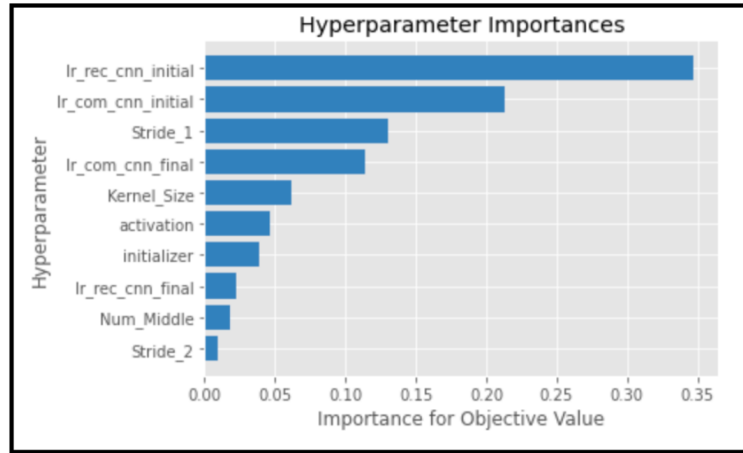


Figure 5.2: 8-Bit EO Model Optimization Hyperparameter Importances Plot

are experimented with resulting in a total of 32 combinations. 96 trials are performed to account for variance, which at 50 epochs takes 150 hours. This study is lengthy due to convolution hyperparameters being changed increasing the size of the model. The best model came as a surprise, vastly outperforming Jian with a PSNR of 40.69, however the CR is significantly lower. The model has 16 middle layers, a Com and Rec CNN stride of 1, and kernel size of 1. These results are suspicious since a stride of 1 for the ComCNN meant the image is not being downscaled. This led to the creation and utilization of the testing notebook to validate actual model performance.

Table 5.2 below shows the performance of the supposed most optimized model against the baseline Jian model. Even though the PSNR is much higher, the CR is almost a third

of the Jian model. It should be noted CRs are much higher in testing due to the calculation

Model	PSNR	CR	SSIM
Lay/Str/Ker Model	40.05	3.01	0.9798
Jian Model	29.68	10.30	0.8461

Table 5.2: 8-Bit EO Optimized Model Testing Results

occurring before CONNECT resizes the image, whereas this calculation occurs after in training. The value in testing is a better representation of CR and shows the Jian model still outperforms in compression. Additionally, the visual perception shown below in Figure 5.3 shows the vast difference in PSNR is not that noticeable to the naked eye. The lack of

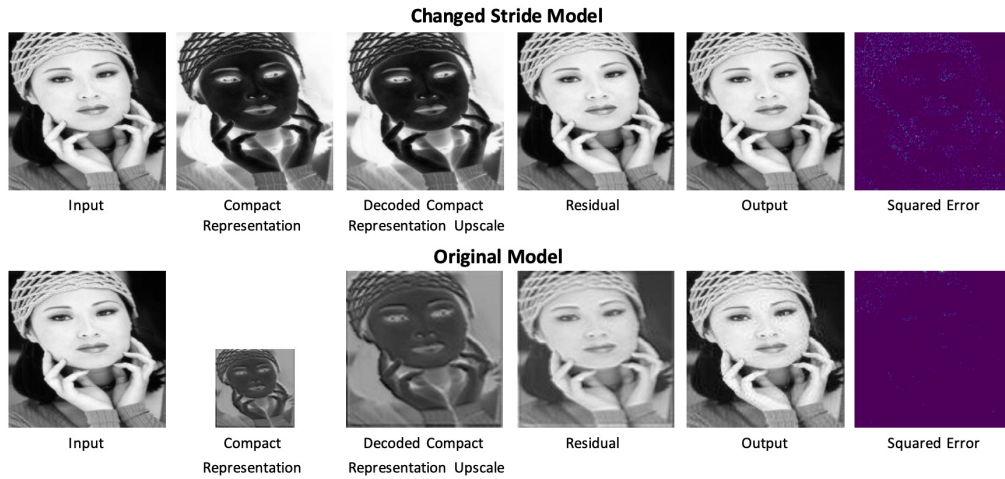


Figure 5.3: 8-bit EO Visual Perception from Optimized and Original Model

compression performed by the model is also visible in the compact representation. With a stride and kernel size of 1, the model effectively removes the ComCNN from the equation.

The most optimized model is hence deemed to be a tie between the activation function/initializer model and the Jian model. Since the Jian model is the baseline and the difference in results negligible, it is utilized as the optimized model for QF analysis and OD.

5.2.2 14-bit IR

14-bit IR is the next datatype to be evaluated following the successful implementation of Optuna and model optimization of the 8-bit EO datatype. Since all hyperparameters to be studied are identified, the Optuna study can be performed all at once. Additionally, the hyperparameters deemed as optimal, ComCNN/RecCNN lr and ComCNN strides, are frozen to reduce the size of the study space. The criteria for model selection is the best 8 models based on PSNR, CR and SSIM.

An initial small training study is performed with a 20 percent batch size or 1650/8800 images on the ADAS FLIR training dataset. The hyperparameters studied are the number of middle layers for RecCNN: (14 – 24), kernel size: (1 – 7), activation function: ReLU, Elu, Tanh, Softmax, initializer: He_normal, Xavier and strides for Rec CNN: (1, 2), At 380 trials and 20 epochs per trial, each takes 40 to 70 minutes for a total study length of 7 days. No trials are pruned as a result of utilizing a grid sampler instead of a pruner. This process ensures minimal variance with three trials per parameter combination. The results and best models from this small training study can be seen below in Table 5.3.

Some notable analyses from the study are that the combinations of a kernel size of 7 and 1 with greater than 18 middle layers generally perform above baseline. Additionally, the Xavier initializer outperforms He_normal, with all other activation functions outperforming

Model	Activation Function	Initializer	Kernel Size	Number of Middle Layers	Stride RecCNN	PSNR	CR	SSIM
#1	Elu	He_normal	1	22	1	59.98	2.85	0.9345
#2	Tanh	Xavier	7	24	1	57.48	2.70	0.9356
#3	Elu	Xavier	7	20	1	57.07	2.80	0.9340
#4	ReLU	He_normal	3	24	2	54.57	4.31	0.9243
#5	ReLU	Xavier	7	20	2	54.72	4.31	0.9342
Jian Model	ReLU	He_normal	3	18	1	40.04	3.17	0.9421
#7	ReLU	He_normal	3	24	1	47.05	3.14	0.9432
#8	Tanh	He_normal	3	14	1	50.21	3.11	0.9412

Table 5.3: 14-Bit IR Model Optimization Small Training Results

Softmax. The number of middle layers and kernel size are the most important parameters according to the importance chart in Figure 5.4.

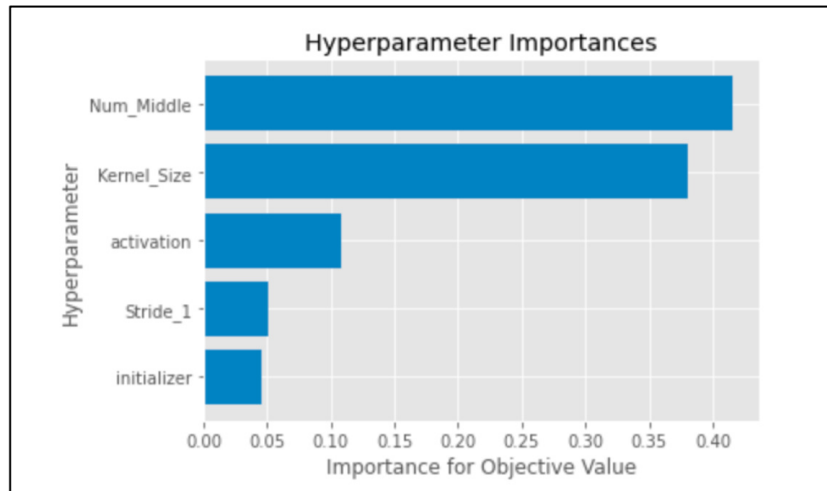


Figure 5.4: 14-Bit IR Model Optimization Hyperparameter Importances Plot

The 8 best performing models are fully trained with 100 percent batch size for 100 epochs, taking 3 days on two GPUs. Table 5.4 shows models with kernel sizes of 1 and

7 outperforming the baseline Jian model in quality and compression metrics. The testing notebook is then utilized to verify the actual performance of the models.

Model	Activation Function	Initializer	Kernel Size	Number of Middle Layers	Stride RecCNN	PSNR	CR	SSIM
#1	Elu	He_normal	1	22	1	60.23	2.48	0.9386
#2	Tanh	Xavier	7	24	1	56.01	2.57	0.9365
#3	Elu	Xavier	7	20	1	54.84	2.95	0.9377
#4	ReLU	He_normal	3	24	2	57.94	4.21	0.9256
#5	ReLU	Xavier	7	20	2	57.67	4.20	0.9231
Jian Model	ReLU	He_normal	3	18	1	58.33	2.82	0.9498
#7	ReLU	He_normal	3	24	1	59.09	2.67	0.9545
#8	Tanh	He_normal	3	14	1	58.21	2.53	0.9522

Table 5.4: 14-Bit IR Model Optimization Large Training Results

Table 5.5 shows the testing results of the 8 best models from the 14-bit IR study. The baseline established by the Jian model is a PSNR of 48.97, CR of 12.89 and SSIM of 0.9966. Model 1 outperforms in PSNR, but significantly under-performs in CR and SSIM. Model 2 outperforms the Jian model in PSNR, CR with a smaller SSIM value. Models 3, 4 and 5 all return unusable results as a stride of 2 and kernel size of 7 is seemingly problematic. Finally, model 7 outperforms Jian in PSNR, CR with almost identical SSIM whereas model 8 outperforms in PSNR, but under-performs in CR and SSIM. Upon first glance of the quality and compression metrics, it seems model 2 is the most optimized, but the visual perception provides another story. In Figure 5.5, model 2 has block artifacts whereas models 7 and 8 represent the input and Jian baseline much better.

The best training PSNR values result from kernel sizes of 1 and 7 but is not representative in testing results with 100 testing images. A kernel size of 1 is problematic as it leads to overfitting and reduces the depth of the network to the where point the model won't learn

Testing Models	PSNR	CR	SSIM
#1	51.06	10.93	0.9937
#2	50.95	14.39	0.9939
#3	-48.14	16.33	0.8462
#4	null	null	null
#5	null	null	null
Jian Model	48.97	12.89	0.9966
#7	50.49	13.32	0.9965
#8	50.15	12.46	0.9964

Table 5.5: 14-bit IR Optimized Model Testing Results

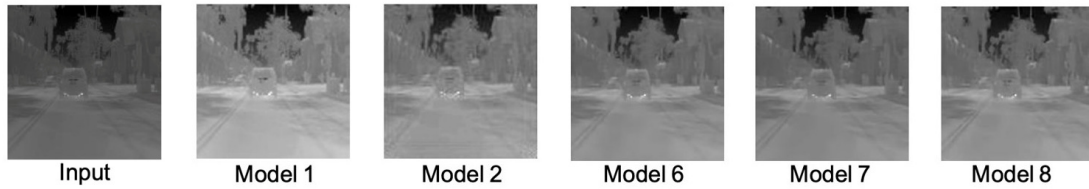


Figure 5.5: 14-bit IR Visual Perception from Optimized Models

properly. Additionally, the smaller kernel size doesn't allow for local feature extraction with no information from neighboring pixels. A kernel size of 7 is also problematic as it can lead to NaN values and a larger kernel size reduces detail in smaller features. A stride size of 2 is also incompatible with testing even though some of the best training values had a stride 2. The movement of 2 pixels at a time reduces image size in the RecCNN, making it unusable.

The most optimized model is determined to be model 7 which has the same general parameters as Jian, with a larger number of middle layers. This is due to the larger 14-bit images requiring a deeper network for better performance. The resulting optimized model faces many hurdles to be utilized any further in OD. The YOLOv5 model only takes 8-bit inputs and proves to be too complex to change. Additionally, the JPEG2000 image codec

is utilized for training and testing due to its larger bit depth. This leads to inconsistencies in as most of the thesis focuses on base JPEG. Since the implementation into OD is too complex, 14-bit IR is not pursued any further for QF analysis or OD.

5.2.3 8-bit IR

The final datatype analyzed for model optimization is 8-bit IR which has the simplest implementation. This is due to the Optuna pipeline being fully developed, with ideal hyperparameters and experimental conditions flushed out. A multi-objective studied is performed as to target the compression and quality metrics more effectively. This eliminates the hyperparameter importance plot, which is irrelevant due to the most important parameters identified in the 16-bit IR study.

The hyperparameters selected for the 8-bit IR study are number of middle layers of the RecCNN: 14 - 24, kernel size: 3, 5, activation function: ReLU, Softmax, Tanh, Elu, and initializer: He_normal, Xavier. 96 total trials are performed at 30 epochs per trial, with a 25 percent batch size. On two Roland GPUs the total study time is 281 hours or slightly less than 6 days. The criteria for small training model selection is more specific than prior

Model	Activation Function	Initializer	Number of Middle Layers	Kernel Size	PSNR	CR	SSIM
Model 1	ReLU	Xavier	16	5	33.90	2.34	0.858
Model 2	Softmax	He_normal	14	3	33.86	2.37	0.857
Jian Model	ReLU	He_normal	18	3	29.18	2.71	0.750

Table 5.6: 8-Bit IR Model Optimization Small Training Results

studies with best models needing a PSNR greater than 33, CR greater than 2.3, and SSIM

greater than 0.85. Two models met the base criteria seen in Table 5.6 with greater PSNR and SSIM, but lesser CR than the Jian model parameters.

These two models are then fully trained at 100 epochs with a 90 percent batch size. This study takes about 60 hours on two Roland GPUs or about two and a half days. The results from the large training results validated that of the small training study seen in Table 5.7. With fully validated training results, the models are evaluated through the testing

Model	Activation Function	Initializer	Number of Middle Layers	Kernel Size	PSNR	CR	SSIM
Model 1	ReLU	Xavier	16	5	34.33	2.32	0.873
Model 2	Softmax	He_normal	14	3	33.34	2.34	0.862
Jian Model	ReLU	He_normal	18	3	32.78	2.40	0.849

Table 5.7: 8-Bit IR Model Optimization Large Training Results

notebook. Testing reveals the Jian model parameters still outperform Models 1 and 2 in compression metrics but under-performs in quality metrics. The visual perception output

Model	PSNR	CR	SSIM
Model 1	29.28	6.84	0.826
Model 2	29.30	6.57	0.830
Jian Model	28.92	7.05	0.815

Table 5.8: 8-Bit IR Optimized Model Testing Results

from the testing notebook did not return any substantial differences between the models. The outputs generated were all very similar with the only notable difference being from the residual of model 1. Hence, model 1 is selected as superior between the top two models.

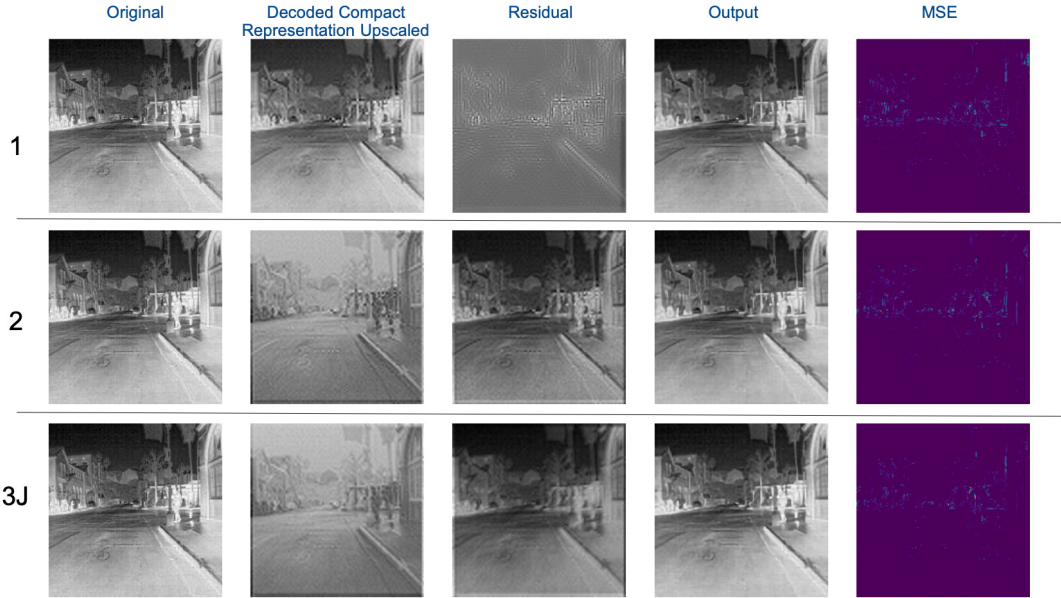


Figure 5.6: 8-bit IR Visual Perception from Optimized Models

Since the Jian model performed better in CR, it is also evaluated alongside model 1 in QF analysis and OD.

5.3 Quality Factor Analysis of Optimized Model

This section details the QF analysis performed on models optimized by Optuna. JPEG image codec QF results are compared against optimized models to see where they perform against the state of the art compression algorithm.

Each optimized model has its weights saved to expedite the training process. Optuna is utilized to automate the training and testing of each saved model on all 21 JPEG QF parameters from 0 to 100 in steps of five. Optuna is also used outside of the DL compression framework to automate the selection of these QF on a JPEG testing notebook. The JPEG

and optimized model/s QF testing results are overlayed and compared in PSNR vs CR plots.

5.3.1 8-bit EO

For 8-bit EO, the Jian hyperparameters are deemed to be optimal and tested against the JPEG image codec. In this subsection, the Jian hyperparameters are referred to as the CONNECT or Jian model. Figure 5.7 shows the output of JPEG QF on the left and CONNECT model QF on the right. Each QF plot compares the CR in the x-axis against the PSNR in the y-axis. From these plots, JPEG is seen to have a parabolic correlation between

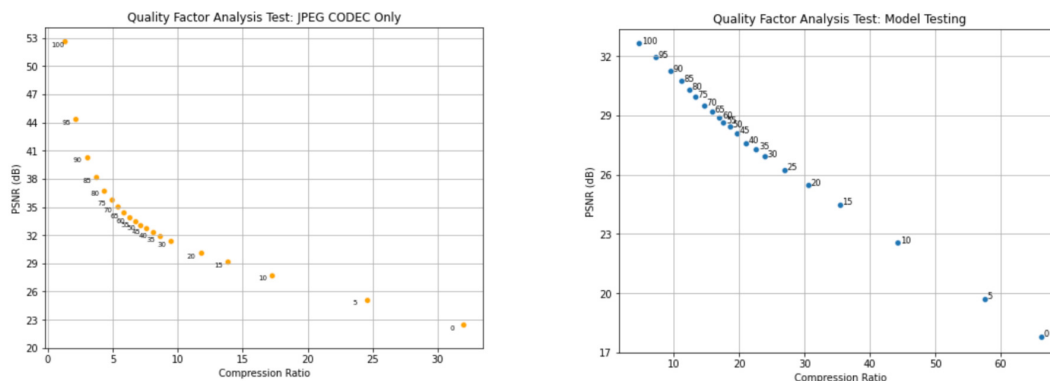


Figure 5.7: 8-bit EO QF Analysis Testing Plots for JPEG and CONNECT

CR and PSNR whereas the CONNECT model's correlation is linear. Additionally, JPEG reaches a max PSNR of 55 for higher QF, whereas CONNECT only achieves a maximum PSNR of 33. However, CONNECT reaches a higher CR max of 67 for a lower QF compared to a JPEG max of 35. The intersection between the JPEG codec and CONNECT model QF is plotted in Figure 5.8. From this plot, it should be noted that CONNECT does not outperform JPEG on a one-to-one QF basis for PSNR. The Jian paper recognized this and

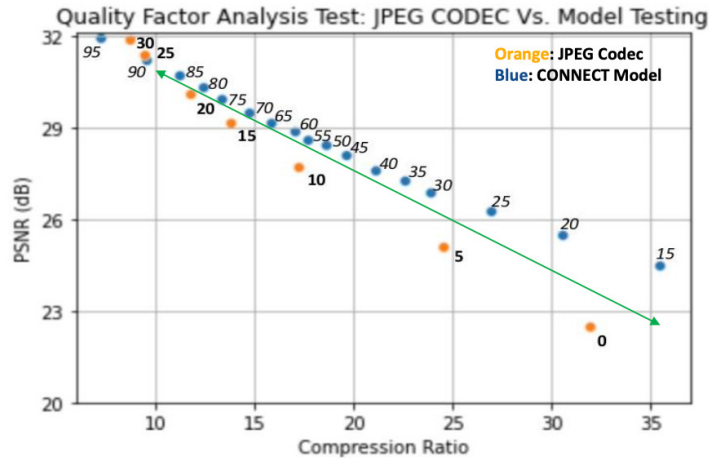


Figure 5.8: 8-bit EO QF Analysis Testing Plot Overlay for JPEG vs CONNECT

compared JPEG QF 5/10 to whichever comparable QF in their model. However, based on Figure 5.8, it can be surmised for all QF values, the CONNECT model outperforms 23 percent of JPEG codec QF values. This is determined as the percent of CONNECT QF above the line of intersection. Additionally, Figure 5.9 shows the CONNECT model QF values of 10-17 outperform the maximum CR of JPEG while maintaining a higher PSNR.

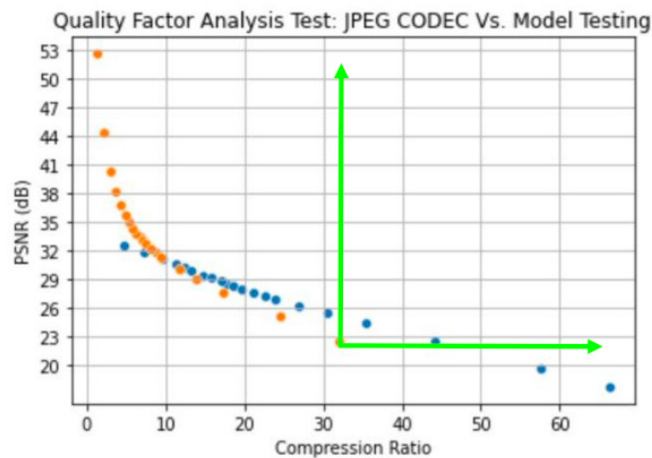


Figure 5.9: 8-bit EO QF Compression Plot Results for JPEG vs CONNECT

5.3.2 8-bit IR

For 8-bit IR, two models are compared against the JPEG image codec. The optimized model of the 8-bit IR datatype is referred to as the Optuna model whereas the Jian parameters are referred to as the Jian model. These two models are overlayed and compared against the JPEG image codec on a QF basis in a similar methodology to 8-bit EO.

The QF analysis in this subsection evaluates two different image sizes as well. In the testing notebook, images are resized to all have the same 180 x 180 size. This allows for consistent results however, non-resized images were experimented with to see the effect of larger image sizes on the optimized models. Figure 5.10 shows the difference resizing images have on JPEG. Non-resized or larger images yield better PSNR and CR results for all models. This is because larger images allow for deeper compression and higher quality.

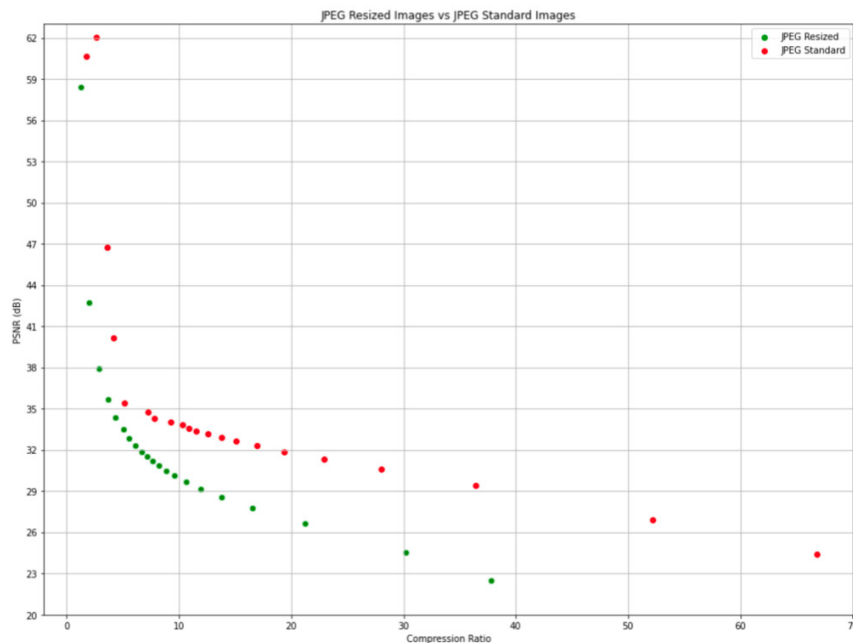


Figure 5.10: 8-bit IR QF Resized vs Non-Resized Results for JPEG

The Optuna and Jian model are plotted against the JPEG codec for non-resized image QF values in Figure 5.11. To evaluate model performance, the linear intersection between each model and JPEG is determined. The Optuna Model intersects at a JPEG QF value of 42 and the Jian Model intersects at a JPEG QF value of 35. This implies for all QF values the Optuna Model is greater than 39 percent of JPEG codec QF values with the Jian Model greater than 32 percent of JPEG codec QF values.

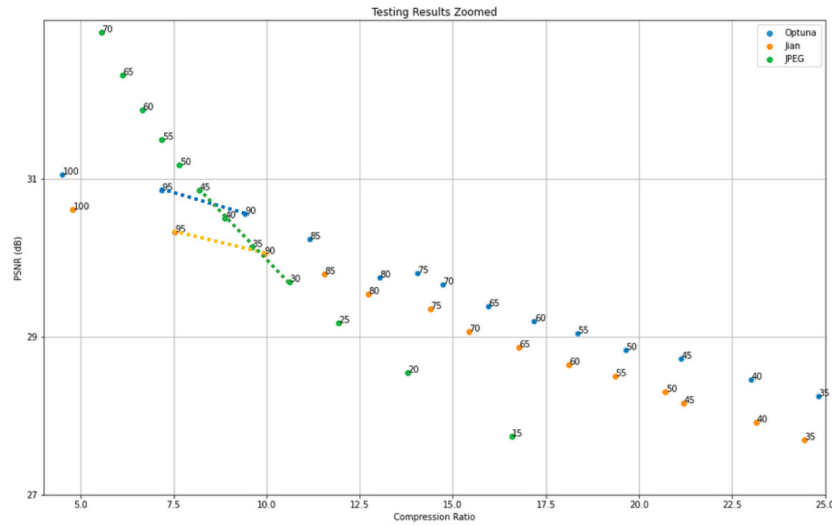


Figure 5.11: 8-bit IR QF Resized Model Testing Results of JPEG vs Optimized Models

Overall the CONNECT framework performs much better with smaller resized images than bigger non-resized images against JPEG. The Optuna and Jian model are plotted against the JPEG codec for resized image QF values in Figure 5.12. The Optuna Model intersects at a JPEG QF value of 12 and the Jian Model intersects at a JPEG QF value of 7. This implies for all QF values the Optuna Model is greater than 4 percent of JPEG codec QF values with the Jian Model greater than 2 percent of JPEG codec QF values. While

these values are low it should be noted with larger images the optimized models begin to break the linear trend and are very close to matching JPEG.

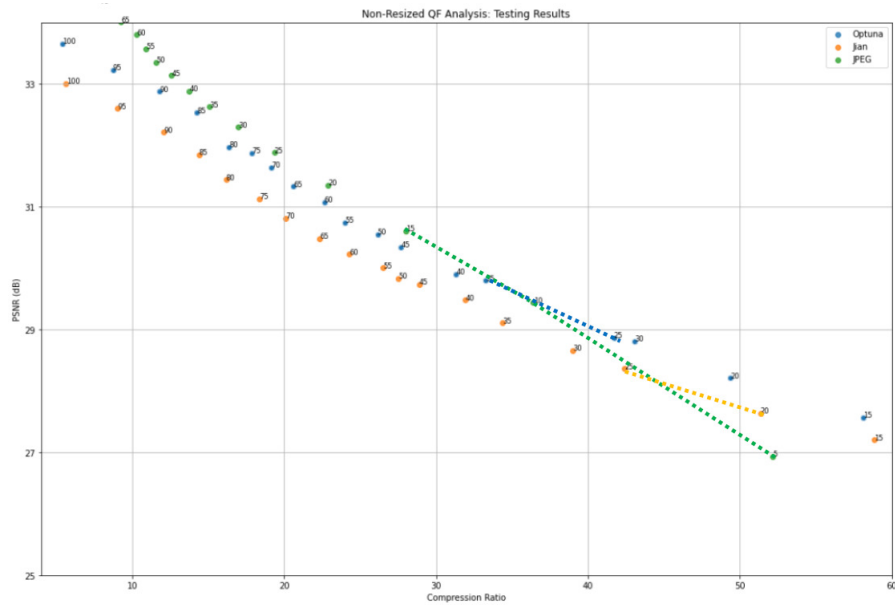


Figure 5.12: 8-bit IR QF Non-Resized Model Testing Results of JPEG vs Optimized Models

In a similar fashion to the 8-bit EO compression results in Figure 5.9, the models can outperform JPEG PSNR at max compression.

The Optuna and Jian model QF values of 10-17 outperform the maximum CR of JPEG while maintaining a higher PSNR for resized images.

Additionally, the Optuna and Jian model QF values of 5-12 outperform the maximum CR of JPEG while maintaining a higher PSNR for non-resized images.

5.4 Object Detection Analysis

The final results section details the application of optimized CONNECT models to the YOLOv5 OD network. The COCO [29] and ADAS FLIR [30] OD datasets are utilized to train 8-bit EO and IR datatypes respectively. For experimentation, 8-bit EO is analyzed based on the QF parameters experimented in the Jian et al. [25] paper. 8-bit IR is analyzed against a wider QF parameter search along with different image sizes. QF baseline charts, image metrics, detection metrics and OD visualizations are provided to give a well rounded perspective of the overall results.

5.4.1 8-bit EO

The optimized Jian model is utilized from 8-bit EO hyperparameter optimization to train images for the YOLOv5 OD network. Additionally, the JPEG codec testing notebook output images are also applied to the YOLOv5. To replicate experimental conditions from the Jian paper, a QF of 10 is applied to base JPEG and a QF of 60 is applied to the Jian model. The QF value of 60 is a result of the performance of the Jian papers model in relation to JPEG. This correlation can be seen in Figure 5.13 where the model and JPEG achieve the same CR.

The results of OD testing with these QF is seen in Table 5.9. The image quality and compression metrics show CONNECT outperforming in SSIM, CR and BPP while achieving a similar PSNR. In addition, CONNECT outperforms JPEG in all OD categories with precision under-performing slightly. Most importantly, CONNECT has a mAP score of 0.365 compared with JPEG's mAP score of 0.121. The mAP score is the most important

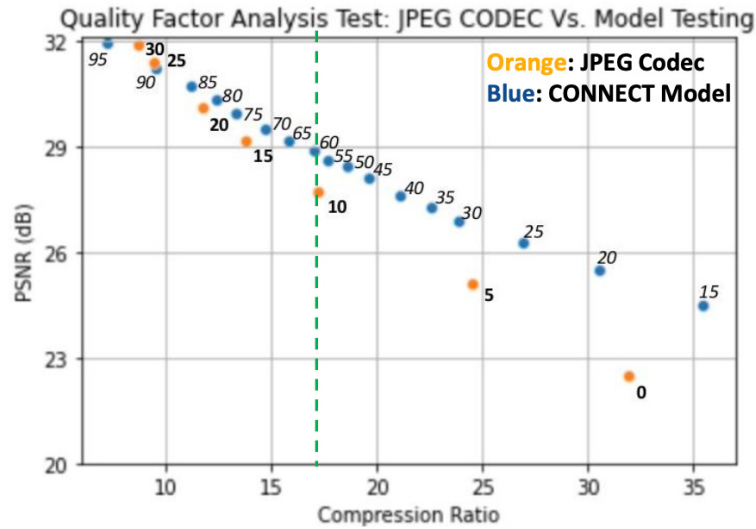


Figure 5.13: 8-Bit EO Detection QF Baseline Plot

metric as it is a combination of all OD metrics while also taking into account the results of the bounding boxes.

Image Quality and Compression Metrics

	PSNR	SSIM	CR	BPP
CONNECT	26.974	0.791	31.202	1.123
JPEG	27.219	0.774	28.822	0.316

Object Detection Metrics

	Accuracy	Precision	Recall	F1 - Score	mAP
CONNECT	0.401	0.664	0.503	0.572	0.365
JPEG	0.197	0.699	0.215	0.329	0.121

Table 5.9: 8-Bit EO Detection and Image Metric Results Chart

The success of the CONNECT framework is also seen in Figure 5.14 via the 8-bit OD output images. The CONNECT framework with Jian parameters at a QF of 60 gets

10/13 detections and 2/2 detections in the first and second images respectively. The JPEG codec gets 3/13 detections and 0/2 detections in the first and second images respectively. In summary, the CONNECT model improves detection AND compression while achieving similar PSNR.

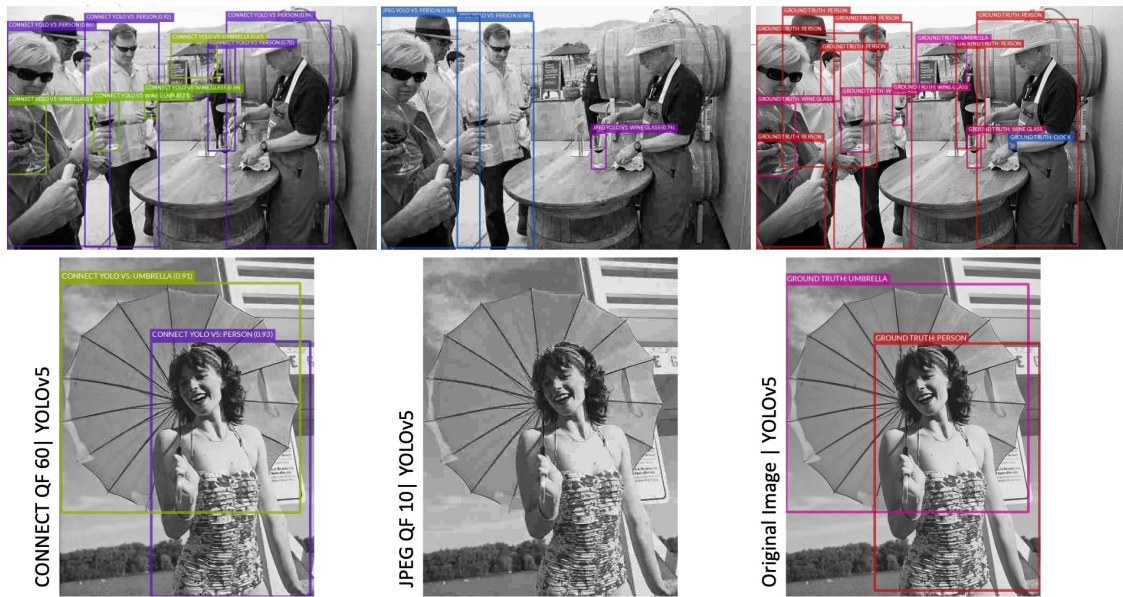


Figure 5.14: 8-Bit EO Detection Visualization Output with Bounding Boxes

5.4.2 8-bit IR

The 8-bit IR OD analysis is more in depth than the 8-bit EO due to progression from the Optuna optimization process. The ADAS FLIR [30] dataset is utilized to train the YOLOv5 network and optimized CONNECT framework models. The optimized Optuna model and Jian model are compared against base JPEG. The image quality and compression metrics alongside the OD metrics are compared at the standard JPEG QF of 95. Additionally, A

QF analysis similar to that of subsection 5.3.2 is performed for both resized and non-resized testing images for all models.

Figure 5.10 displays a comparison amongst the Optuna model, Jian model and JPEG codec for a universal QF of 95. As expected, the JPEG codec outperforms both models in PSNR and SSIM, under-performing in CR by almost half. As for OD results, the mAP score is most important with JPEG yielding a score of 0.303 outperforming the Optuna model's 0.275 and Jian model's 0.274.

Image Quality and Compression Metrics

Model	PSNR	SSIM	CR
JPEG	42.51	0.842	3.45
Optuna	29.28	0.826	6.84
Jian	28.92	0.815	7.05

Object Detection Metrics

Model	Precision	Recall	F-1 Score	mAP
JPEG	0.83	0.82	0.83	0.303
Optuna	0.86	0.76	0.81	0.275
Jian	0.86	0.76	0.81	0.274

Table 5.10: 8-Bit IR Detection and Image Metric Results Chart

The 8-bit IR detection results visualization in Figure 5.15 provide a different outlook than the metrics. All three outputs have a perfect 5/5 detections with the main discrepancies coming from the confidence score in each detection. This implies at a QF of 95, the Optuna and Jian models can achieve similar detections to Jian at a higher CR.

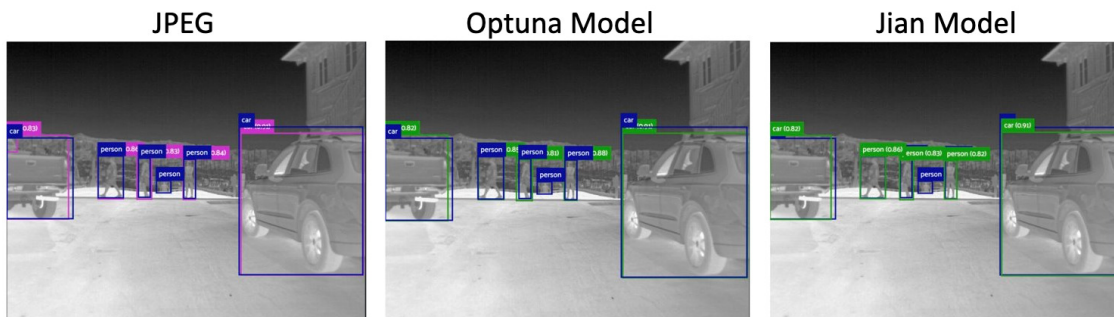


Figure 5.15: 8-Bit IR Detection Visualization Output with Bounding Boxes

These observations were taken a step further with a full QF analysis run on both resized and non-resized images for Jian, Optuna and JPEG. However, the resized images proved to be fruitless as resizing the images drastically reduces detection rates. The metrics were so low across the board there was nothing worth of noting. JPEG vastly outperforms both models, but the mAP scores were still very low and detections almost non-existent. In this experiment, the Optuna model outperforms the Jian model for a majority of QF values.

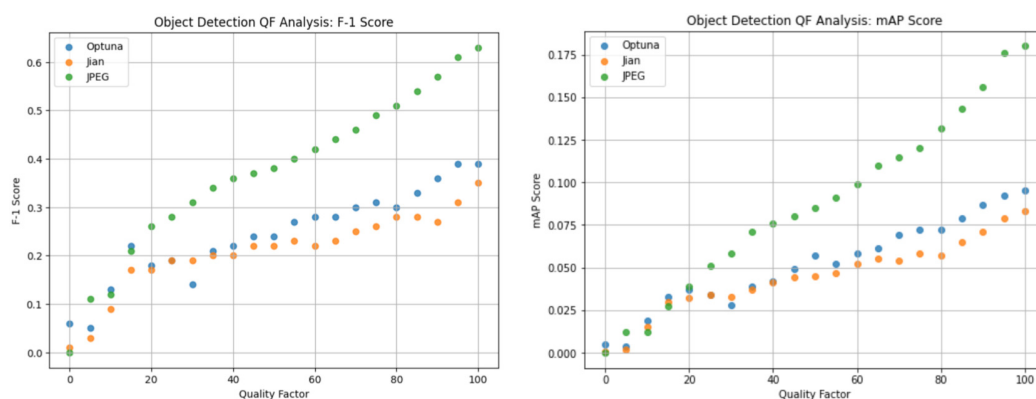


Figure 5.16: 8-Bit IR Resized Detection QF Results of JPEG vs Optimized Models

The best QF results from OD came from the non-resized image analysis where a baseline was established. This baseline of a mAP score greater than or equal to 0.30 and f-1 score greater than 0.80 determines where the CONNECT model gets similar detection metrics to JPEG. According to the non-resized detection QF results in Figure 5.17, Jian and Optuna match detections for a QF greater than or equal to 95, validating initial findings. It should be not Jian outperforms for QF values less than 95, but Optuna outperforms where it matter for QF greater than 95. In summary, optimized CONNECT models can match JPEG detections and outperform CR for a QF of 95.

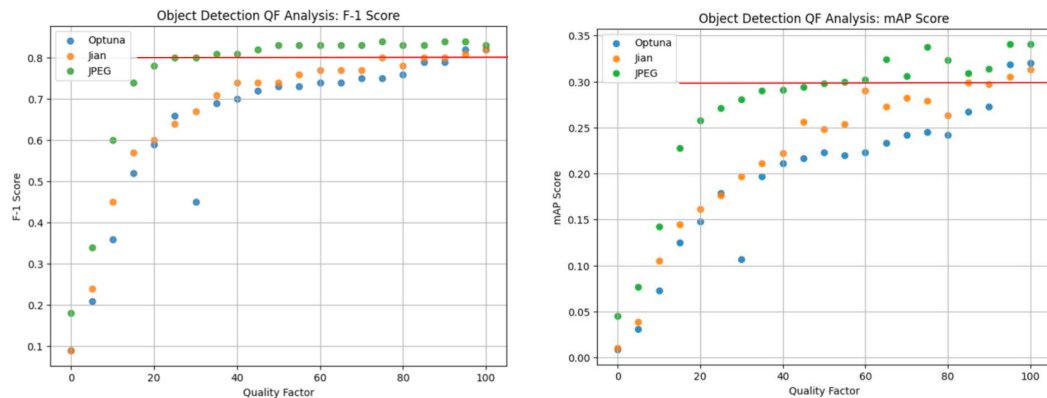


Figure 5.17: 8-Bit IR Non-Resized Detection QF Results of JPEG vs Optimized Models

CHAPTER VI

FUTURE WORK AND CONCLUSION

The final chapter details future work that can be performed to improve the project and summarizes the major findings of the thesis.

6.1 Future Work

There are several different characteristics of the CONNECT framework that can be improved upon to further explore its capabilities. This section lays out the main categories and paths the project can take in the future.

6.1.1 New Image Data Types

The main datatypes studied in this thesis are that of IR and EO visible data. These datasets are primarily made up of terrestrial images and scratch the surface of publicly available datasets. The next step is to train the framework on large scale data and more complex image types.

Satellite imagery is one category of a large scale datatype that can heavily benefit from efficient compression. Satellite data comes in IR, EO, multi-spectral and hyper-spectral datatypes. Hyper-spectral and multi-spectral satellites collect and process information from across the electromagnetic spectrum [55]. These are utilized to develop topological maps and complex land use images. The applications for DL compression in the field of satellite imagery are broad as the need to increase communication speed increases.

6.1.2 Modernized Image Codecs

JPEG is heavily studied in the CONNECT project as it has been the state-of-the-art compression algorithms for decade. However, lossy compression can have some downsides and better codecs are emerging. JPEG2000 is a notable contender as it is lossy and outperforms JPEG but falls short of PNG for non-photographic images [4].

JPEG XL is proposed as the next generation image codec to be utilized as it outperforms both JPEG and PNG. This image codec is designed by the same JPEG team that developed JPEG and JPEG 2000. The new image codec is effective with compression at perceptual qualities that range from visual nearly lossless to mathematically lossless [56]. The CONNECT framework could see vast improvement across all datatypes with this codec integrated into the pipeline.

6.1.3 Performance Evaluation Metrics

As the image compression metrics were very straight forward, the quality metrics of PSNR and SSIM had some visual shortcomings. It is proposed in addition to PSNR and SSIM, new quality metrics be implemented for evaluation.

A new evaluation metric called gradient magnitude similarity deviation (GMSD) is an efficient image quality assessment model. Since image gradients are sensitive to distortions, local structures suffer from degradation. GMSD utilizes pixel-wise gradient magnitude similarity between reference and distorted images to accurately predict perceptual image quality [57]. Another evaluation metric to be utilized in tandem with GMSD is the Universal Quality Index. Instead of using traditional MSE, the index similar to GMSD models image distortion with three factors. These are the loss of correlation, luminance distortion

and contrast distortion which assist the index in performing better than MSE[58]. The implementation of both these metrics alongside PSNR and SSIM will help provide a well rounded picture in evaluating the framework.

6.1.4 16-bit in Object Detection

A major obstacle in 16-bit IR object detection was YOLOv5 not supporting 16-bit image conversions. A solution is to utilize TensorFlow Lite as it support converting activations to 16-bit integer values and weights to 8-bit integer values during model conversion [59]. The TensorFlow team refers to this as 16x8 quantization mode which can improve the quality of a quantized model significantly. Additionally, this would allow higher bit-depth images to be converted into a format to be used by lower bit-depth models. TensorFlow light models benefit from super-resolution, image de-noising, and HDR reconstruction from a single image. Introducing TensorFlow Light to the YOLOv5 network could bring 16-bit object detection into the project pipeline.

6.2 Conclusion

The objective of this thesis was to implement an optimization process into an existing framework and test it's success on object detection network. This goal was met with resounding success as models were optimized for three different datatypes and tested on the YOLOv5 detection network.

The initial Jian parameters of the model were deemed fully optimal for 8-bit EO data and provided a proof of concept for further optimization. A deeper model was optimized for the larger bit depth 14-bit IR datatype revealing Jian was only optimized for EO data.

Additionally, the Jian model is far outperformed in quality metrics by the optimized Optuna model for 8-bit IR data.

The QF analysis took optimization a step further and shed light on where the CONNECT framework models outperformed the base JPEG codec. For 8-bit IR resized data, CONNECT matched JPEG PSNR and CR outputs. The Optuna Model is greater than 39 percent of JPEG codec QF values with the Jian Model greater than 32 percent of JPEG codec QF values. For 8-bit IR non-resized data, CONNECT matched JPEG PSNR and CR outputs to a lesser extent. The Optuna Model is greater than 4 percent of JPEG codec QF values with the Jian Model greater than 2 percent of JPEG codec QF values. In the 8-bit EO domain the Jian model outperformed 23 percent of JPEG codec values. Additionally, the CONNECT framework outperformed the PSNR and CR at max compression of JPEG. For IR resized, both the Optuna and Jian models outperformed PSNR values at max QF for a QF of 0 to 17. For IR non-resized, both the Optuna and Jian models outperformed PSNR values at max QF for a QF of 5 to 12. Finally, EO resized outperformed JPEG at a QF of 10-17. Not only was the CONNECT framework full optimized for all datatypes studied, but was proven to be comparable in many instances to a state of the art compression algorithm.

In the application of OD, the optimized CONNECT models achieved similar detection metrics to JPEG at a baseline mAP score greater than 0.30 and f-1 score greater than 0.80. This was achieved for Jian and Optuna models at a QF greater than or equal to 95. Not only can the CONNECT models match JPEG detections but also outperform CR for a QF greater than or equal to 95.

Model testing results reveal the success of CONNECT framework optimization for 8-bit EO, 14-bit IR and 8-bit IR. The subsequent models optimized in each datatype were also

shown to perform well in the realm of OD. The results are very promising and validate the work done to optimize the DL compression framework.

BIBLIOGRAPHY

- [1] K. Sayood, *Introduction to data compression*. MK Morgan Kaufmann Publissers, an imprint of Elsevier, 2018.
- [2] G. Hudson, A. Léger, B. Niss, and I. Sebestyén, “Jpeg at 25: Still going strong,” *IEEE MultiMedia*, vol. 24, no. 2, pp. 96–103, 2017.
- [3] E. Öztürk and A. Mesut, “Performance evaluation of jpeg standards, webp and png in terms of compression ratio and time for lossless encoding,” *2021 6th International Conference on Computer Science and Engineering (UBMK)*, pp. 15–20, 2021.
- [4] C. Christopoulos, A. Skodras, and T. Ebrahimi, “The jpeg2000 still image coding system: an overview,” *IEEE Transactions on Consumer Electronics*, vol. 46, no. 4, pp. 1103–1127, 2000.
- [5] M. Khanafer and S. Shirmohammadi, “Applied ai in instrumentation and measurement: The deep learning revolution,” *IEEE Instrumentation Measurement Magazine*, vol. 23, no. 6, pp. 10–17, 2020.
- [6] P. Ongsulee, “Artificial intelligence, machine learning and deep learning,” *2017 15th International Conference on ICT and Knowledge Engineering (ICT KE)*, pp. 1–6, 2017.
- [7] S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a convolutional neural network,” *2017 International Conference on Engineering and Technology (ICET)*, pp. 1–6, 2017.
- [8] J. Yang and J. Li, “Application of deep convolution neural network,” *2017 14th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, pp. 229–232, 2017.
- [9] L. Deng, “The mnist database of handwritten digit images for machine learning research,” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.
- [10] A. Krizhevsky, V. Nair, and G. Hinton, “Cifar-10 (canadian institute for advanced research),” 2009. [Online]. Available: <http://www.cs.toronto.edu/~kriz/cifar.html>
- [11] R. Chauhan, K. K. Ghanshala, and R. Joshi, “Convolutional neural network (cnn) for image detection and recognition,” *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, pp. 278–282, 2018.
- [12] J. S. Lim, *Two-Dimensional Signal and Image Processing*. Englewood Cliffs, NJ: Prentice Hall, 1990.
- [13] Y. Wan and Q. Xie, “A novel framework for optimal rgb to grayscale image conversion,” *2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, vol. 02, pp. 345–348, 2016.

- [14] V. Janani and M. Dinakaran, “Infrared image enhancement techniques — a review,” *Second International Conference on Current Trends In Engineering and Technology - ICCTET 2014*, pp. 167–173, 2014.
- [15] T. Jinno, M. Okuda, and N. Adami, “Detail preserving multiple bit-depth image representation and coding,” *2011 18th IEEE International Conference on Image Processing*, pp. 1533–1536, 2011.
- [16] L. Pető and J. Botzheim, “Parameter optimization of deep learning models by evolutionary algorithms,” *2019 IEEE International Work Conference on Bioinspired Intelligence (IWOBI)*, pp. 000 027–000 032, 2019.
- [17] B. Qolomany, M. Maabreh, A. Al-Fuqaha, A. Gupta, and D. Benhaddou, “Parameters optimization of deep learning models using particle swarm optimization,” *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 1285–1290, 2017.
- [18] H. Choi and I. V. Bajic, “High efficiency compression for object detection,” *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1792–1796, 2018.
- [19] Toderici, O’Malley, Hwang, Vincent, Minnen, Baluja, Covell, and Sukthankar, “Variable rate image compression with recurrent neural networks,” *arXiv*, 2015.
- [20] G. Toderici, D. Vincent, N. Johnston, S. J. Hwang, D. Minnen, J. Shor, , and M. Covell, “Full resolution image compression with recurrent neural networks,” *arXiv*, 2016.
- [21] L. Theis, W. Shi, A. Cunningham, and F. Husza, “Lossy image compression with compressive autoencoders,” *arXiv*, 2017.
- [22] J. Balle, V. Laparra, and E. P. Simoncell, “End-to-end optimized image compression,” *arXiv*, 2016.
- [23] M. Li, W. Zuo, S. Gu, D. Zhao, and D. Zhang, “Learning convolutional networks for content-weighted image compression,” *arXiv*, 2017.
- [24] G. Jocher, “Yolov5 network,” 2022. [Online]. Available: <https://github.com/ultralytics/yolov5>
- [25] F. Jiang, W. Tao, S. Liu, J. Ren, X. Guo, and D. Zhao, “An end-to-end compression framework based on convolutional neural networks,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 3007–3018, 2018.
- [26] H. Talebi, D. Kelly, X. Luo, I. G. Dorado, F. Yang, P. Milanfar, and M. Elad, “Better compression with deep pre-editing,” *IEEE Transactions on Image Processing*, vol. 30, pp. 6673–6685, 2021.

- [27] G. Yang, W. Feng, J. Jin, Q. Lei, X. Li, G. Gui, and W. Wang, "Face mask recognition system with yolov5 based on image recognition," *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pp. 1398–1404, 2020.
- [28] J. Ieamsaard, S. N. Charoensook, and S. Yammen, "Deep learning-based face mask detection using yolov5," *2021 9th International Electrical Engineering Congress (iEECON)*, pp. 428–431, 2021.
- [29] L. et al., "Coco (microsoft common objects in context)," 2014. [Online]. Available: <https://paperswithcode.com/dataset/coco>
- [30] T. F. LLC, "Free teledyne flir thermal dataset for algorithm training," 2022. [Online]. Available: <https://www.flir.com/oem/adas/adas-dataset-form/>
- [31] L. A. J. Abel, T. C. N. Oconer, and J. C. Dela Cruz, "Realtime object detection of pantry objects using yolov5 transfer learning in varying lighting and orientation," *2022 2nd International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, pp. 1–7, 2022.
- [32] A. Horé and D. Ziou, "Image quality metrics: Psnr vs. ssim," *2010 20th International Conference on Pattern Recognition*, pp. 2366–2369, 2010.
- [33] Z. Wang and A. C. Bovik, "Mean squared error: Love it or leave it? a new look at signal fidelity measures," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98–117, 2009.
- [34] R. Patel, V. Kumar, V. Tyagi, and V. Asthana, "A fast and improved image compression technique using huffman coding," *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pp. 2283–2286, 2016.
- [35] D. R. Setiadi and I. Moses, "Psnr vs ssim: imperceptibility quality assessment for image steganography," *Multimedia Tools and Applications*, vol. 80, no. 6, pp. 8423–8444, 2021.
- [36] A. F. Gad, "Mean average precision (map) explained," Apr 2021. [Online]. Available: <https://blog.paperspace.com/mean-average-precision>
- [37] B. Mirza, D. Haroon, B. Khan, A. Padhani, and T. Q. Syed, "Deep generative models to counter class imbalance: A model-metric mapping with proportion calibration methodology," *IEEE Access*, vol. 9, pp. 55 879–55 897, 2021.
- [38] A. M. F. Zarth and T. B. Ludermir, "Optimization of neural networks weights and architecture: A multimodal methodology," *2009 Ninth International Conference on Intelligent Systems Design and Applications*, pp. 209–214, 2009.
- [39] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," 2019. [Online]. Available: <https://arxiv.org/abs/1907.10902>

- [40] “A hyperparameter optimization framework.” [Online]. Available: <https://optuna.readthedocs.io/en/stable/>
- [41] X. Bouthillier, P. Delaunay, M. Bronzi, A. Trofimov, B. Nichyporuk, J. Szeto, N. Sepah, E. Raff, K. Madan, V. Voleti, and et al., “Accounting for variance in machine learning benchmarks,” Mar 2021. [Online]. Available: <https://arxiv.org/abs/2103.03098>
- [42] H. V. Pham, S. Qian, J. Wang, T. Lutellier, J. Rosenthal, L. Tan, Y. Yu, and N. Nagappan, “Problems and opportunities in training deep learning software systems: An analysis of variance,” *2020 35th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 771–783, 2020.
- [43] A. Senior, G. Heigold, M. Ranzato, and K. Yang, “An empirical study of learning rates in deep neural networks for speech recognition,” *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6724–6728, 2013.
- [44] D. Arora, M. Garg, and M. Gupta, “Diving deep in deep convolutional neural network,” *2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, pp. 749–751, 2020.
- [45] S. Maitra, R. K. Ojha, and K. Ghosh, “Impact of convolutional neural network input parameters on classification performance,” *2018 4th International Conference for Convergence in Technology (I2CT)*, pp. 1–5, 2018.
- [46] D. Chanda, Ali-Emam-Al-Badi, and A. K. M. N. Islam, “Performance analysis of initialization algorithms of deep neural network based coordinated beamforming system for mmwave,” *2021 5th International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*, pp. 1–6, 2021.
- [47] R. Parhi and R. D. Nowak, “The role of neural network activation functions,” *IEEE Signal Processing Letters*, vol. 27, pp. 1779–1783, 2020.
- [48] “Linux: Modern enterprise open source.” [Online]. Available: <https://ubuntu.com/>
- [49] “Jupyterlab: A next-generation notebook interface.” [Online]. Available: <https://jupyter.org/>
- [50] “Python software foundation.” [Online]. Available: <https://www.python.org/>
- [51] “An end-to-end open source machine learning platform.” [Online]. Available: <https://www.tensorflow.org/>
- [52] “Pytorch: From research to production.” [Online]. Available: <https://pytorch.org/>
- [53] D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics,” pp. 416–423, July 2001.

- [54] M. Bevilacqua, “Set5,” 2014. [Online]. Available: <https://paperswithcode.com/dataset/set5>
- [55] M. L. Clark, “Mapping land cover with hyperspectral and multispectral satellites using machine learning and spectral mixture analysis,” *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 513–516, 2016.
- [56] “Time for the next-gen codecs to dethrone jpeg.” [Online]. Available: <https://cloudinary.com/blog>
- [57] W. Xue, L. Zhang, X. Mou, and A. C. Bovik, “Gradient magnitude similarity deviation: A highly efficient perceptual image quality index,” *IEEE Transactions on Image Processing*, vol. 23, no. 2, pp. 684–695, 2014.
- [58] Z. Wang and A. Bovik, “A universal image quality index,” *IEEE Signal Processing Letters*, vol. 9, no. 3, pp. 81–84, 2002.
- [59] “Post-training integer quantization with int16 activations.” [Online]. Available: <https://www.tensorflow.org/lite>