EMPLOYMENT OF CYCLE-SPINNING IN DEEP LEARNING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

ÜLKÜ UZUN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF
DOCTOR OF PHILOSOPHY
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

NOVEMBER 2022

EMPLOYMENT OF CYCLE-SPINNING IN DEEP LEARNING

submitted by **ÜLKÜ UZUN** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Information Systems Department, Middle East Technical University** by,

Date: 29.11.2022

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

Name, Surname: ÜLKÜ UZUN

Signature :

ABSTRACT

EMPLOYMENT OF CYCLE-SPINNING IN DEEP LEARNING

UZUN, ÜLKÜ
Ph.D., Department of Information Systems
Supervisor: Prof. Dr. Alptekin Temizel

November 2022, 102 pages

Cycle-spinning (CS) method has been used in wavelet domain processes such as signal denoising, and image enhancement with great success. In this thesis, CS is adapted to be used in deep-learning algorithms, particularly it is integrated into GAN-based raindrop removal and CNN based image classification, and object detection models. Experiments on commonly-used architectures, such as AlexNet, DenseNet, ResNet, YOLOv5, EfficientDet, CenterNet, and TensorFlow Object Detection (TF-OD) show that the application of the CS method produces favorable results with higher perceptual quality in terms of full-reference metrics for raindrop removal, increased accuracy in classification and better object detection performance. It is shown that the proposed method reduces the signal degradation caused by aliased components when it is employed before down-sampling and it can increase the performance, without introducing any extra learnable parameters. Another advantage of the CS method is that it inherently provides a smoothing-out effect and, as such, it has potential uses in denoising.

Keywords: Cycle-spinning, Generative Adversarial Network (GAN), Classification, Object detection, Shift-invariant, Aliasing

ÖZ

DERİN ÖGRENMEDE DÖNGÜLÜ ÇEVİRİM YÖNTEMİNİN KULLANIMI

UZUN, ÜLKÜ
Doktora, Bilişim Sistemleri Bölümü
Tez Yöneticisi: Prof. Dr. Alptekin Temizel

Kasım 2022, 102 sayfa

Döngülü-Çevirim (DÇ) yöntemi, sinyal gürültü giderme ve görüntü iyileştirme gibi dalgacık alanı süreçlerinde büyük başarı ile kullanılmaktadır. Bu tezde DÇ yöntemi, derin öğrenme algoritmalarında kullanılmak üzere uyarlanmış, özellikle GAN tabanlı yağmur damlası giderme modellerine ve CNN tabanlı görüntü sınıflandırma ve nesne algılama modellerine entegre edilmiştir. AlexNet, DenseNet, ResNet, YOLOv5, EfficientDet, CenterNet ve TensorFlow Object Detection (TF-OD) gibi yaygın olarak kullanılan mimariler üzerinde yapılan deneyler, DÇ yönteminin yağmur damlalarının giderilmesine etkisi referans bazlı metriklerle değerlendirildiğinde daha yüksek algısal kalite verdiğini göstermektedir. Sınıflandırmada ve nesne tespiti problemlerinde ise doğruluk ve daha yüksek nesne tespiti algılama performansı görülmüştür. Önerilen yöntemin, alt-örnekleme (downsampling) aşamasından önce kullanıldığı takdirde ekstra öğrenilebilir herhangi bir parametreye ihtiyaç duymadan örtüşme (aliasing) nedeniyle ortaya çıkan sinyal bozulmalarını azalttığı ve genel performansı arttırabileceği gösterilmiştir. DÇ yönteminin bir başka avantajı da doğası gereği bir yumuşatma

etkisi sağlaması ve bu nedenle gürültü gidermede potansiyel kullanımlara sahip olmasıdır.

Anahtar Kelimeler: Döngülü-Çevirim (DÇ), Çekişmeli Üretici Ağlar, Sınıflandırma, Nesne tanıma, Kayma değişmezliği, Örtüşme

To my family...

ACKNOWLEDGMENTS

First and foremost, I am extremely grateful to my supervisor, Prof. Dr. Alptekin TEMİZEL for his invaluable advice, continuous support, and patience during my Ph.D. study. His immense knowledge and plentiful experiences have encouraged me all the time in my academic research and daily life.

I would also thank my thesis committee for their valuable comments and suggestions. I would also thank Prof.Dr.A.Aydın ALATAN for his suggestions that made my research so much richer.

I would like to express my gratitude to my family. Without their tremendous understanding and encouragement over the past few years, it would be impossible for me to complete my study.

A special feeling of gratitude to my loving mother and father, Hasbiye and Şaban UZUN whose words of encouragement and push for tenacity ring in my ears. My sister Selcan UZUN ALTUNCU, her husband Uğraş ALTUNCU, and my little sunshine Serra ALTUNCU have never left my side and are very special.

Finally, I am also grateful to my many friends who have supported me throughout the process. I will always appreciate what they do, especially Assoc. Prof. Dr. M. Yılmaz ÜSTÜNER for helping me improve my technology skills, editing for hours, and being my constant supporter.

TABLE OF CONTENTS

ABSTRACT iv

ÖZ vi

DEDICATION viii

ACKNOWLEDGMENTS ix

TABLE OF CONTENTS x

LIST OF TABLES xiv

LIST OF FIGURES xv

LIST OF ABBREVIATIONS xviii

CHAPTERS

1 INTRODUCTION 1

1.1 Problem Definition 3

1.2 Contributions of the Study 4

1.3 Organization of the Thesis 5

2 RELATED WORK 7

2.1 Cycle-spinning (CS) method 7

2.2 Deep learning algorithms 11

2.3 Convolutional Neural Networks (CNNs) 11

2.3.1 Image classification using CNNs 14

2.3.2 Object detection using CNNs 14

2.4 Generative Adversarial Networks (GANs) 17

2.4.1 Image enhancement using GANs 18

3 METHODOLOGY 21

3.1 Cycle-spinning (CS) method 21

3.1.1 Integrating CS into GAN-based models 23

3.1.2 Integrating CS into the convolutional layer of CNN-based models 23

3.2 Shifting 35

4 INTEGRATING CS INTO GAN-BASED MODELS FOR RAINDROP REMOVAL 39

4.1 DeRaindrop algorithm 39

4.1.1 Datasets 39

4.1.2 Architecture 40

4.1.3 Evaluation metrics 41

4.1.4 Results 44

5 INTEGRATING CS INTO CNN-BASED MODELS FOR IMAGE CLASSIFICATION 49

5.1 AlexNet 50

5.1.1 Datasets 50

5.1.2 Architecture 51

5.1.3 Evaluation Metrics 51

5.1.4 Results 51

5.2 DenseNet 54

5.2.1 Architecture 54

5.2.2 Results 55

5.3 ResNet 56

5.3.1 Dataset 56

5.3.2 Architecture 57

5.3.3 Results 57

6 INTEGRATING CS INTO CNN-BASED MODELS FOR OBJECT DETECTION 59

6.1 YOLOv5 59

6.1.1 Datasets 60

6.1.1.1 Cheetah-Human thermal dataset 60

6.1.1.2 COCO (COmmon Objects in Context) dataset 60

6.1.1.3 Dataset for Object Detection in Aerial Images (DOTA) . 61

6.1.2 Architecture 62

6.1.3 Evaluation metrics 62

6.1.3.1 Intersection Over Union (IOU) 62

6.1.3.2 Precision and Recall 63

6.1.3.3 F1-Score 63

6.1.3.4 Mean Average Precision (mAP) 64

6.1.3.5 MS-COCO Method 64

6.1.4 Results 65
6.1.4.1 Yolov5 Cheetah-Human dataset results 65
6.1.4.2 YOLOv5 COCO dataset results 68
6.1.4.3 Yolov5 DOTA dataset results 70
6.2 EfficientDet 76
6.2.1 Dataset selection 76
6.2.2 Architecture 76
6.2.3 Results 77
6.2.3.1 EfficientDet COCO Dataset results 77
6.3 CenterNet 79
6.3.1 Dataset selection 79
6.3.2 Architecture 79
6.3.3 Results 80
6.3.3.1 CenterNet Visdrone Dataset results 80
6.4 Discussion 81
7 CONCLUSION AND FUTURE WORK 85
REFERENCES 89

LIST OF TABLES

Table 1 Average full- and no-reference evaluation results on raindrop added real-life images and object detection measure mAP results. 44

Table 2 AlexNet Accuracy Results for CIFAR10 Dataset Without Augmentation. 52

Table 3 AlexNet Accuracy Results for CIFAR10 Dataset With Augmentation. 53

Table 4 DenseNet Accuracy Results for CIFAR10 Dataset Without Augmentation. 55

Table 5 DenseNet Accuracy Results for CIFAR10 Dataset With Augmentation. 56

Table 6 Accuracy Results of ResNet101 trained for ImageNet-Contrast 50000. 58

Table 7 The detection performance of the CS integrated YOLOv5s on the Cheetah-Human dataset using different shift amounts. 65

Table 8 Detection performance of Yolov5s-CS on COCO validation set using different shift amounts. 69

Table 9 Detection performance of YOLOv5s, YOLOv5s-CS_1, CS_2, CS_{1+2} on COCO validation set. 69

Table 10 Detection performance of CS integrated YOLOv5s on DOTA validation set using different shift amounts. 71

Table 11 Detection performance of YOLOv5s, YOLOv5s-CS_1, CS_2, CS_{1+2} on DOTA validation set. 72

Table 12 Detection performance of EfficientDet, EfficientDet-CS_1, CS_2, CS_{1+2} on COCO validation set. 77

Table 13 Detection performance of CenterNet, CenterNet-CS_1, CS_2,CS_{1+2} on VISDRONE2019-DET validation set. 81

Table 14 Computational complexity results at 512x512 input resolution. 82

LIST OF FIGURES

Figure 1 Representation of one shift of JPEG. 8

Figure 2 Shifts are employed for various directional CS types [1]. 10

Figure 3 CNN architectures over a timeline. 12

Figure 4 Milestones of object detection. (Adapted from [2]) 15

Figure 5 The CS block diagram illustrates the adaptation of the CS method by wrapping the algorithm. 24

Figure 6 Illustration of the cycle spinning (CS) method implementation. . . 25

Figure 7 Illustration of the cycle spinning (CS) method implementation detail. 25

Figure 8 Output channels of a standard convolution operation on input and element-wise multiplication of matrices followed by a max-pooling are shown for the illustration. 26

Figure 9 Convolution results of f(t), g(t) functions without the CS method employed. 27

Figure 10 Convolution results of f(t), g(t) functions after the employment of the CS method. 27

Figure 11 The basic model for analyzing the results of 2D image processes. 27

Figure 12 Visual comparison of pooling results on synthetic test patterns. . . 29

Figure 13 Original image of Lena (left) and noisy version (right) with noise. 30

Figure 14 Visual comparison of pooling results on noisy Lena image. 31

Figure 15 Integrated gradient visualization results of the model on CIFAR-10 test images. 33

Figure 16 Overview of Grad-CAM heatmap. (Adapted from [3]) 33

Figure 17 Comparison of the GradCAM visualizations. 34

Figure 18 Shifting examples of an image (Generated by Keras ImageData-Generator Class). 35

Figure 19 Examples of images with border objects that are not fully in the frame. 36

Figure 20 The representation of four different padding methods on a sample image from the CIFAR-10 dataset. 37

Figure 21 Architecture of the attentive generative adversarial network [4]. . 41

Figure 22 NIQE workflow diagram. 43

Figure 23 BRISQUE workflow diagram. 43

Figure 24 BLIINDS-II workflow diagram. 44

Figure 25 Demonstration of a test image, raindrop applied version, and raindrop removal results. 45

Figure 26 Demonstration of a natural scene test image, raindrop applied version, and raindrop removal results. 47

Figure 27 Demonstration of a test image, raindrop applied version, and raindrop removal results. 48

Figure 28 Object detection scores of (a) ground truth images, (b) images with raindrops, (c) DeRaindrop algorithm output, (d) 1-shift CS output. 48

Figure 29 Block diagram of DenseNet classification algorithm. 54

Figure 30 Precision-Recall (PR) graphs of the Cheetah-Human validation dataset. 66

Figure 31 F1 graphs of the Cheetah-Human validation dataset. 67

Figure 32 Object detection results on the Cheetah-Human test set using baseline YOLOv5s (left) and YOLOv5s-CS_1 (right). 68

Figure 33 The Cheetah-Human test image class label detail. 68

Figure 34 COCO test set results using base YOLOv5s model (left) and YOLOv5s-CS_1 (right). 70

Figure 35 Comparison of the original YOLOv5s and YOLOv5s-CS_1 version (mAP@0.5) on the DOTA test dataset. 72

Figure 36 DOTA test set results using base YOLOv5s model and YOLOv5s-CS_1. 73

Figure 37 Precission-Recall (PR) graph of the YOLOv5s (top) and YOLOv5s-CS_1 trained on the DOTA dataset. 74

Figure 38 F1 graph of the YOLOv5s (top) and YOLOv5s-CS_1 trained on the DOTA dataset. 75

Figure 39 COCO test dataset results using EfficientDet base model and EfficientDet-CS_1. 78

Figure 40 Comparison of the original and CS_{1+2} YOLOv5s in terms of mAP@0.5 values of objects in the VisDrone2019-DET validation dataset. 80

LIST OF ABBREVIATIONS

CS	Cycle-Spinning
GAN	Generative Adversarial Network
CNN	Convolutional Neural Network
CIFAR	Canadian Institute for Advanced Research
YOLO	You Only Look Once
DWT	Decimated Wavelet Transform
JPEG	Joint Photographic Experts Group
WZP	Wavelet-domain Zero Padding
PCS	Partial Cycle Spinning
UWT	Undecimated Wavelet Transform
RCS	Reduced Cycle Spinning
ANN	Artificial Neural Network
DNN	Deep Neural Network
VGG	Visual Geometry Group
ILSVRC	Large Scale Visual Recognition Challenge
COCO	Common Objects In Context
HR	High Resolution
LR	Low Resolution
HF	High Frequency
MNIST	Modified National Institute of Standards and Technology
GPU	Graphics Processing Units
PSNR	Peak Signal-to-Noise Ratio

SSIM	Structural Similarity Index Measure
IQA	Internal Quality Assurance
NIQE	Naturalness Image Quality Evaluator
BRISQUE	Blind/Referenceless Image Spatial Quality Evaluator
DCT	Discrete Cosine Transform
API	Application Programming Interface
SSD	Single Shot Detection
IOU	Intersection Over Union
mAP	MEAN Average Precision
AP	Average Precision
DOTA	Dataset for Object deTection in Aerial images
FPN	Feature Pyramid Network
NAS	Neural Architecture Search
NAS-FPN	Neural Architecture Search-Feature Pyramid Network
BİFPN	Bi-directional Feature Network

CHAPTER 1

INTRODUCTION

Although the theoretical background of current deep learning applications is based on perceptron studies in the 1950s, the field has accelerated with recent developments in hardware, software, and data science. Today, neural networks based deep learning algorithms such as Convolutional Neural Networks (CNNs), Generative Adversarial Networks (GANs), Recurrent Neural Networks (RNNs) have opened a new horizon.

Convolutional Neural Networks (CNNs) are among the most prominent models in deep learning and are frequently employed for image recognition and classification. Research on CNNs is still ongoing, while improvements of these networks are becoming more and more challenging due to problems to overcome physical and computational barriers. On the other hand, due to their ubiquitous use, even small improvements in their performance are desirable and have real-world impact.

Despite being considered mainly shift and rotation invariant, CNNs performance may be adversely affected by small pixel changes [5] and small shifts, translations, and rotations can reduce its accuracy [6]. Because the combination of convolution and pooling operations hampers their invariance to small shifts (i.e., a few pixels) in position and other minor transformations of an object in modern deep learning algorithms. It has been shown that even a single-pixel shift may alter the overall result [7, 8]. Furthermore, blurring and additive noise have also detrimental effects on the accuracy of CNN-based algorithms [9]. This is especially important in deep learning-based systems because artifacts on an image may adversely affect various stages of deep learning methods. It can be claimed that these disadvantages might be overcome by employing traditional artifact removal methods such as applying various filtering

techniques. However, it can be advantageous to use more robust models against noisy data, as image quality may vary after the deployment [5]. For this reason, instead of using a pre-processing stage to clean the noisy images, it is more desirable to have methods that work more reliably when fed such images.

This dissertation's main focus is the adaptation of the cycle-spinning (CS) method to use in deep networks and evaluate its impact on performance. CS method has originally been used in association with the wavelet transform to remove noise and artifacts in images with notable performance. On the other hand, its integration into deep learning algorithms has not been explored before. In the CS method, the data is first shifted by a small amount (i.e., a few pixels), the shifted data are then denoised, the denoised data are then unshifted, and finally, the unshifted data are averaged to obtain the final denoised result. The basic idea behind this approach is combining several shifted versions of the processed signal, this produces a high redundancy of the processed information and allows different points of view, resulting in converging to better solutions in wavelet-based denoising. Because convolutional layers in the time domain are equivalent to multiplication in the frequency domain, it is possible to adapt the CS method to use in convolutional layers. The fact that modern convolutional networks are not shift-invariant (to small shifts), as in the wavelet transform, supports our proposal that the employment of the CS method would contribute to obtaining better results in these algorithms. The proposed method does not need any additional trainable parameters and can be straightforwardly integrated into convolution layers without architectural changes. In the scope of this thesis, we integrate CS into deep networks and mainly investigate its application in (i) raindrop removal on images using CS-integrated generative adversarial network (GAN), (ii) image classification using CNNs by integrating CS into a convolutional layer of classification networks, and (iii) object detection by integrating CS into a convolutional layer of object detection networks.

1.1 Problem Definition

Deep learning algorithms have demonstrated notable performance improvements in a number of real-world applications during the past several years. While several improvements lead to better performance, deep learning algorithms are criticized for lacking resilience against various image corruptions [10, 11, 12, 13]. Previous studies have shown that representations learned by pre-training large models with noisy data can be useful for prediction. However, training using noisy images also often results in lower accuracy. Noisy images can naturally occur in real-world contexts due to the use of noisy image sources, image shifts, and a variety of reasons. For this reason, alternative methods are explored that can continuously add new data during training, do not require clean-label data, and easily adapt to standard training pipelines with little computational or memory overhead [14].

More specifically, a lack of invariance to small image deformations was reported in several resources [7, 8, 15, 16]. This property has also been exploited to generate adversarial images with the purpose to fool a network [17, 18]. These studies propose and conclude, that modern networks are not robust against small shift amounts. Currently, various convolution types, such as deformable convolution [19, 20] and dynamic region-aware convolution [21], have been proposed to improve invariance against large and unknown transformations. Deformable convolution with Deformable pooling [19] extends the sampling grid with learnable offsets and spatial sampling grids in the input feature map, instead of getting feature maps at fixed locations in standard convolutions. Similarly, the Adaptive Attention Network (AANet) [20] adaptively adjusts the receptive fields to extract features with an attention-based encoder. Dynamic Region-Aware Convolution (DRConv) [21] changes the spatial feature in order to obtain better model semantic variations and also can maintain translation invariance and extract more plentiful information.

The main components of CNNs are convolutional and pooling layers, followed by non-linearities and subsequent fully-connected layers. The pooling layers effectively subsample the data and one of the consequences of subsampling a signal is the unde-

sired effect of aliasing. Even if appropriate pooling is used, ensuring that a given layer is shiftable, the subsequent nonlinearities in a CNN may not preserve the shift ability, as the nonlinearities may again introduce high frequencies [8]. The Nyquist theorem states that the sampling rate must be at least twice the highest frequency of the signal to avoid aliasing [22]. If the sampling rate is inadequate (i.e., not able to satisfy the Nyquist rate), aliasing may occur. This results in the folding of higher-frequency signals onto the low-frequency portion of the original signal causing distortions and artifacts. As there are no explicit anti-aliasing filters in place at the front, the pooling layer has the potential to cause aliasing. Aliasing is prevented in traditional image/signal processing by using a low-pass filter before subsampling [23], however, this filtering may cause some information loss by removing the high-frequency data [24]. Accordingly, in this work, we propose the integration of cycle-spinning (CS) into the convolution layers of a CNN network. The CS method, used in the context of the wavelet domain, compensates for the lack of shift-invariance for small shifts in the sampled wavelet transform by averaging over denoised cyclically-shifted versions of the signal or image [25]. Besides its de-noising property, CS also has an anti-aliasing effect by spreading sharp image features across their neighboring pixels. This improves structural similarity between subsampled outputs of an image and its shifted version [26]. Motivated by these observations, we propose integrating the cycle-spinning (CS) into CNN-based models. When integrated into CNN-based models, it does not increase the number of learnable parameters and it does not require any specific augmentation. Combining several shifted versions of the processed signal without aliasing allows an enriched information flow to the subsequent layers.

1.2 Contributions of the Study

In this dissertation, we claim that the CS method, which is actually used for image denoising and enhancement in wavelet-based studies, can also be used in the field of deep learning and its edge recovery and anti-aliasing properties eventually increase the accuracy. The proposed method preserves the semantic properties of the original image. This is in contrast to traditional methods such as denoising the input image

where the original semantic properties are usually lost. The main contributions of this study are as follows:

1) CS is integrated into a GAN for denoising and its performance improvement for raindrop removal is demonstrated.

2) Anti-aliasing effect and edge-sharpening properties of CS are demonstrated in CNNs.

3) It is shown that CS can be integrated into convolutional layer of CNN-based classification networks to improve image classification performance.

4) It is shown that CS can be integrated into a convolutional layer of a CNN-based object detection network to object detection performance.

The work presented in this thesis has led to the following publications [27, 28]:

- Ü. Uzun and A. Temizel, "Cycle-Spinning GAN for Raindrop Removal from Images", 16th IEEE Int. Conf. Advanced Video and Signal Based Surveillance (AVSS), 2019, pp. 1-6, doi: 10.1109/AVSS.2019.8909824
- Ü. Uzun and A. Temizel, "Cycle-Spinning Convolution for Object Detection", IEEE Access, vol. 10, pp. 76340-76350, 2022, doi: 10.1109/ACCESS.2022.3192022

1.3 Organization of the Thesis

The dissertation is organized as follows; Chapter 2 provides a summary of related works on deep learning and cycle-spinning (CS) method. Section 2.1 briefly explains the underlying CS method and its variations used in this research. The deep learning structure is discussed in section 2.2 which provides necessary background information for adapting the CS method into deep learning algorithms. Section 2.3 provides a background on Convolutional Neural Networks (CNNs) and two major applications of CNNs, image classification and object detection are described in Section 2.3.1 and

Section 2.3.2, respectively. Section 2.4 Generative Neural Networks (GANs) aim to give a broader view of the problems and proposed solutions on deep learning-based image processing algorithms focusing particularly on raindrop removal in Section 2.4.1.

Chapter 3 describes the methodology with the experimental design, results, and analysis. This chapter starts with the details of the CS method and its use in imaging applications are provided in Section 3.1. Then a brief introduction to the shifting process which is an important part of the CS method is explained in Section 3.2.

Results and analysis of the experiments are presented in each chapter from Chapter 4 up to Chapter 6. Firstly, we propose an image resolution enhancement algorithm by adapting cycle-spinning (CS) to the raindrop removal problem in hand. The CS method adaptation to the deep learning-based GAN method for the raindrop removal approach is explained in Section 4.1. Section 4.1.1 provides the used dataset. Section 4.1.2 explains the architecture, the experiment design and implementation details, Section 4.1.3 the evaluation metrics details, and Section 4.1.4 the results of the experiments of the proposed method.

Chapter 5 reports our deep learning-based image classification experiments. Section 5.1, Section 5.2, Section 5.3 explains the experimented image classification algorithms. Similarly, Chapter 6 explains deep learning-based object detection algorithm experiments. The used dataset, architectures, evaluation metrics and results of each of the object detection algorithms experienced in Chapter 6 are explained in an organizational structure similar to Chapter 4 and Chapter 5.

Finally, the conclusion and the description of the future work are explained in Chapter 6.

CHAPTER 2

RELATED WORK

Firstly, the cycle-spinning (CS) method is reviewed in this section. Then, although there are various deep learning algorithms available, two prominent algorithms, which are directly related to this thesis, will be discussed: Convolutional Neural Networks (CNNs) and the Generative Adversarial Neural Networks (GANs).

2.1 Cycle-spinning (CS) method

The CS method has been utilized in the processing of different types of images, including radar, astronomical, medical, and classic images. However, CS has also been employed for signal extraction from electrical discharges, denoising in electromagnetic signals or ultrasonic signals [29].

The overall methodology of this dissertation research is based on the employment of the cycle-spinning (CS) method integrated into deep learning algorithms. Note that the CS method has, at the moment of writing, not yet been applied to the deep learning algorithms as our literature survey reveals. We aim to apply the CS method whose effectiveness has already been shown in the wavelet domain to the deep learning-based domain.

In wavelet domain applications, the CS method causes an "averaging out" effect and suppresses visual artifacts caused by decimated wavelet transform (DWT) such as the Gibbs phenomena. The Gibbs phenomenon is a particular behavior of some functions that appear as over and under waves around a jump- discontinuity which is seen

in many scientific problems and applications involving signal and image processing [30]. The input image is shifted, de-noised, and then it is unshifted for a range of shifts. These intermediate results are combined by averaging to obtain a reconstructed image. The resulting image exhibits significantly fewer Gibbs effects than thresholding-based de-noising with the conventional orthogonal wavelet transform.

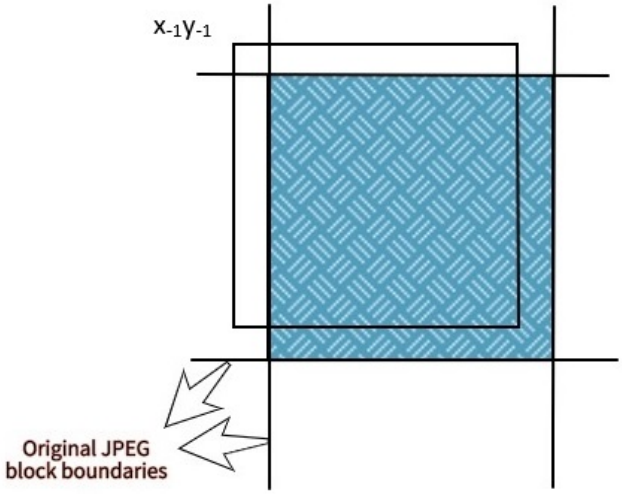


Figure 1: Representation of one shift of JPEG. Notation x and y for spatial shifting is in z-transform which converts a discrete-time signal into a complex frequency domain. (x and y notation show dimension and lower values of x,y show shift amounts, minus shows opposite direction).

Figure 1 demonstrates the shifting process which explains the basic mechanism behind the CS method for JPEG encoding. The JPEG encoding procedure shapes the frequency in borders and this will reduce the intensity of the block appearance by applying different shifts to the JPEG image [31].

It is possible that a signal can be re-aligned to reduce artefacts, but there is no guarantee that this will always be possible. When there are numerous discontinuities in a signal, they may interfere with one another: the optimum shift for one discontinuity may not be ideal for the other. Because of this reason several possible points and their average is recommended for the wavelet domain [32]. The wavelet transform is not time-invariant. It has been shown that, if a noisy signal is displaced and then subjected to denoising, the estimate is different from denoising it without such a shift

[33] [34]. So, the periodic time-invariance of the wavelet transform is used by the CS method.

In [31] and [35], the CS method idea was applied as a post-processing operation after decompression. The results have shown significant improvement in the JPEG and JPEG2000 image compression framework. The CS method has also been shown to be a useful tool for improving the results after wavelet denoising.

The CS method is applied to images in the same manner and the process is summarized in 5 steps below:

- Move the compressed images horizontally and vertically by (i, j)
- Apply JPEG to the shifted image
- Unshift JPEG applied image i.e. vertically and horizontally by (-i,-j)
- Repeat for all possible shifts
- Average all versions

JPEG encoding reduces the high-frequency content of the image and some artificial high-frequency components are added at the block boundaries. Thus the size of data loss reduces [36]. Three different types of shifting operations are offered in Nosratinia's study such as symmetric extension, row, and column replication, and zero-shift replacement. The aforementioned algorithm can be viewed as an approach to reducing artifacts such as blockiness, and ringing artifacts and enhancing visual quality where quantization plays the role of nonlinear denoising.

CS method can be applied to different neighborhood sizes; using a smaller neighborhood suffers from not having adequate information, whereas a larger neighborhood has the risk of crosstalk from spatially uncorrelated image features. However, it should be noted that the choice of the optimum neighborhood size k, is also potentially dependent on image size. Tests demonstrate that the findings indicate that $k = 2$ is a trade-off between computational cost and quality for a variety of image

sizes [1]. The CS method used in this study can be thought as a linear combination of many estimates of the enhanced resolution image.

It has been shown that the CS method yields good results for image denoising and image resolution enhancement [37] [38] [39]. Additionally, the idea of CS was extended by multi-spinning for image denoising [40] which is based on random shifts instead of using known the CS method shift practices such as horizontal, vertical, or diagonal shifts like in Figure 2. The figure indicates the patterns used for CS shift locations used are shown as shaded elements with the no shift location occupying. As a result of this process, they mentioned two advantages averaging avoids Gibb's phenomenon and contribution of quality [40].

In [29], a variant of CS having a reduced computational cost, Partial Cycle Spinning (PCS), is proposed. Although PCS is based on CS, fewer shifts are chosen for this implementation. Also in this study, it is noted that the selection of the shifts has a significant impact on the final denoising outcomes. It is recommended that to select the shifts a sparse way for getting good results with PCS.

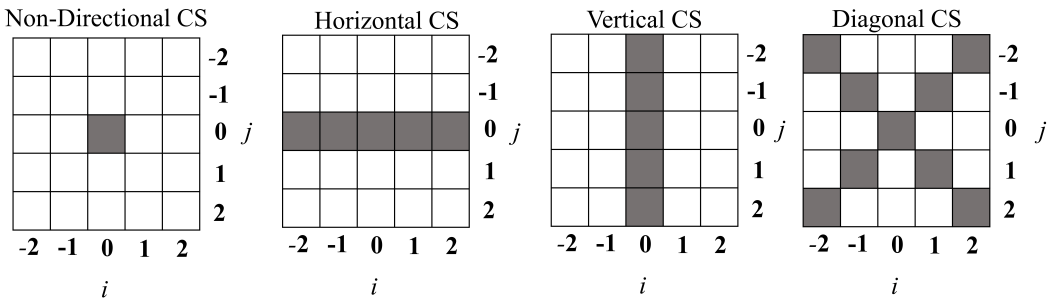


Figure 2: Shifts are employed for various directional CS types [1].

The CS method has certain benefits over other wavelet implementations. Combining several shifted versions of the processed signal produces a high redundancy of the processed information and allows different points of view. Getting values from different points of view can find the best solutions for wavelet denoising problems [38].

2.2 Deep learning algorithms

Deep learning is a subset of machine learning that is loosely inspired by the human brain [41]. Deep learning has gained big popularity and is widely used for solving complex problems [42].

The building block of deep learning algorithms is an artificial neural network (ANN) which consists of artificial neurons, also called nodes. These nodes are stacked next to each other in layers such as the input layer, hidden layer(s), and output layer. Each node multiplies its inputs with random weights, sums up, and adds a bias. Finally, nonlinear functions, also known as activation functions, are applied to the output. If ANNs have more than one hidden layer, they are called Deep Neural Networks (DNNs) and if they are designed specifically with convolutional layers they are called Convolutional Neural Networks (CNNs).

Deep learning algorithms have self-learning representations and during the training process, algorithms use unknown items in input distribution to extract properties, group objects and discover useful data models. Like training machines for self-learning, this happens at multiple levels using algorithms to build models [41]. Deep learning algorithms use different types of neural networks to perform specific tasks and there may be different algorithms better suited to perform specific tasks [43].

Due to its widespread use recently, two fundamental deep learning algorithms, CNNs and GANs, will be employed in this study.

2.3 Convolutional Neural Networks (CNNs)

CNNs, also known as ConvNets, consist of multiple convolutional layers and are mainly used for image processing and object detection. Yann LeCun developed the first CNN in 1988 and it was called LeNet. In this context, the history of deep CNNs can be traced back to the appearance of LeNet [44].

CNNs are among the more popular neural network frameworks that are used in applications like deep learning models for classification [45]. Over the years, considerable amounts of advancements have been done in this area [43]. As shown in Figure 3, there are several different algorithms. According to our literature survey, we have chosen state-of-the-art algorithms to study on. The main breakthrough in CNN performance was brought by AlexNet, which provided a substantial performance leap in 2012-ILSVRC in comparison to conventional computer vision techniques [46].

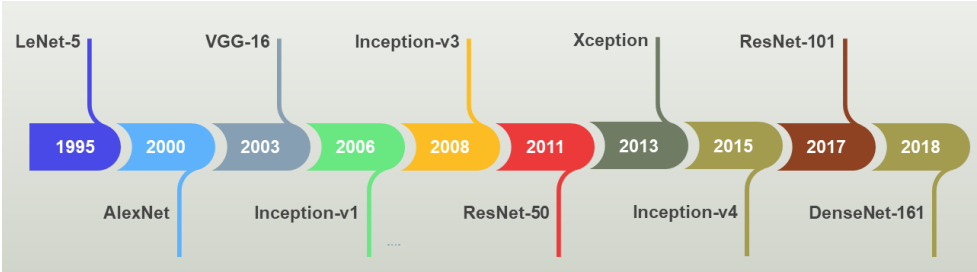


Figure 3: CNN architectures over a timeline.

With the growing popularity of deep CNNs, it is becoming increasingly difficult to establish filter dimensions, stride, padding, and other hyper-parameters for each layer independently. Convolutional layers with a fixed topology that can be replicated several times are used to solve this problem. The paradigm changed from custom layer design to modular and uniform layer design as a result. The principle of modularity in CNNs made them easy to adapt to various tasks [47] [48] [49].

While AlexNet concentrated on 3×3 window sizes and strides in the first convolutional layer, VGG addresses depth. VGG then replaced the 11×11 and 5×5 filters with a stack of 3×3 filters, demonstrating that the small (3×3) filters could cause the effect of a large size filter $(5 \times 5$ and $7 \times 7)$. By reducing the number of parameters, the use of small-size filters gives an added advantage of low computational complexity. Afterward, the Visual Geometry Group (VGG) controls the network's complexity by inserting 1x1 convolutions between the convolutional layers, which also learn a linear combination of the resultant feature maps. Max-pooling is put after the convolutional layer for network tuning, while padding was used to keep the spatial resolution [50].

Later, Inception-V1 was the winner of the 2014-ILSVRC competition and is also known as GoogleNet. The GoogleNet architecture's key goal was to achieve high precision at a low computational cost [51]. Inception-V3, V4, and ResNet are followed by Highway Networks. Srivastava et al. suggested a deep CNN, called Highway Networks, in 2015, based on the intuition that increasing network depth would increase learning ability [52]. Highway Networks were seen to converge considerably faster than simple networks, with 900 layers deep [49].

The Dense Network or DenseNet is also an important milestone in the deep learning architecture timeline. The idea here is that connecting a skip connection from the previous layer improves the performance and this creates a direct route for the information through the network. The research in CNN is still going on and has a significant potential for improvement in the future. Image Classification, Segmentation, Object Detection, Video Processing, Natural Language Processing, and Speech Recognition are some main tasks of CNNs.

CNN is generally known as a good algorithm for its ability to learn by utilizing different stages of information processing. This results in using larger amounts of data but as the technology of data processing has improved, CNN became more popular and related research work has been intensified. According to Khan et.al.; "Several inspiring ideas to bring advancements in CNNs have been explored, such as the use of different activation and loss functions, parameter optimization, regularization, and architectural innovations. However, the significant improvement in the representational capacity of the deep CNN is achieved through architectural innovations" [49]. Various types of layers have been added to CNN to improve its success in recent years. In particular, ideas of using spatial and channel information, the depth and breadth of architecture, and multipath information processing ideas have attracted considerable attention. Similarly, the idea of using a block of layers as a structural unit is also gaining popularity [49].

Besides improvements, studies about the robustness of CNNs to image corruption exist [10, 11, 12, 53]. More specifically, the lack of invariance of modern CNNs to small image deformations was reported in several resources [6, 7, 54] [55]. Even if

appropriate pooling is used, ensuring that a given layer is shiftable, the subsequent nonlinearities in a CNN may not preserve the shift ability, as the nonlinearities may again introduce high frequencies [56]. The Nyquist theorem states that the sampling rate must be at least twice the highest frequency of the signal to avoid overlapping [57]. Modern CNNs are brittle when the input is translated, rescaled, or otherwise transformed. As these studies propose and conclude, modern networks are not shift-invariant.

Accordingly, we offer to add the CS method to convolution neural network layers. Consequently, each pixel learns the surrounding pixel structure and since the average value is taken, stronger learning is to be provided without significant loss of value. Since CNN possesses feature generation and discrimination ability to show the results of our survey we have chosen two prominent CNN capabilities, classification and object detection.

2.3.1 Image classification using CNNs

One of the most well-known CNN applications in computer vision is image classification which consists of classifying an image into one of many different categories [26]. The CNN architecture for classification includes convolutional layers, pooling layers, and fully connected layers. Convolution layers are used to do feature detection, whereas pooling layers are used to select features. Each feature map of a pooling layer is linked to the feature map of the convolutional layer before and lowers the computational burden by reducing the number of connections between convolutional layers. The output from convolution and pooling layers is fed into the fully connected layers for classification [26].

2.3.2 Object detection using CNNs

Object detection aims to detect precise locations and classes of objects in an image. Object detection methods have developed significantly and are used for numerous tasks such as surveillance, disease identification, and autonomous driving.

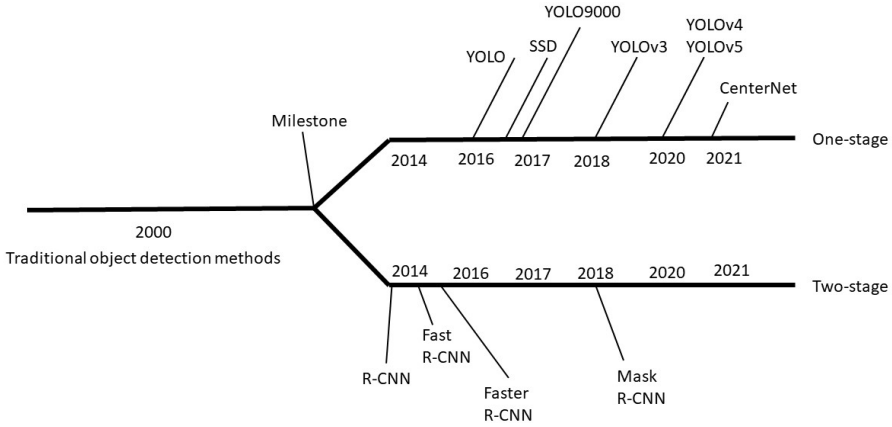


Figure 4: Milestones of object detection. (Adapted from [2])

There are mainly two types of deep learning based object detection approaches: one-stage and two-stage algorithms, as shown in Figure 4. A well-known two-stage based algorithm is Region-CNN (R-CNN) [58], which is based on separating an image into different regions of interest. Afterward, Fast R-CNN and Faster R-CNN [59] have been developed, and in recent years Mask R-CNN [60] has been developed which can illustrate the contours of a target position in addition to the position and type of a target object. One-stage models separate an image into several candidates first and they have higher precision, however higher computational costs limit their use in real-life applications. Therefore, one-stage approach-based algorithms have been proposed. These models perform regression analyses of targets of detection and directly predict the locations of the targets. While it offers faster compute speeds, these method typically have lower accuracy relative to the aforementioned methods as it utilizes single-shot detection to process an image. One-stage object detection models include Single-Shot Detectors (SSD) [61] and the You Only Look Once (YOLO) series (such as YOLO [62], YOLO9000 [63], YOLOv3 [64], YOLOv4 [65], YOLOv5 [66]). The newly released YOLOv4 and YOLOv5 are reported to be more accurate than many two-stage and one-stage approaches. But, the complex network structure of YOLOv4 and YOLOv5 requires higher computational volumes, and consequently, relies on high-level graphics cards, to attain a better learning performance [67]. Ac-

cording to our experiments, YOLOv5 outperforms the YOLOv4 Tiny model when trained on the dataset [68] and evaluated on the same test datasets. As our main interest is the analysis of small shifts, with suitability for the practical application being a primary consideration, YOLOv5 was chosen as the primary object detection algorithm for this study as it offers better speed, detection accuracy, and stability. In addition to YOLOv5, the EfficientDet algorithm [69] has been used to analyze the effects of the CS method. EfficientDet model was developed by Google, and it consistently achieves better efficiency. Similar to Yolov5, EfficientDet has also achieved remarkable performances in Pascal VOC and Microsoft COCO tasks and is widely used in real-world applications. EfficientDet employed state-of-the-art network EfficientNet [70] as its backbone, enabling the model to learn complex objects. EfficientDet, like EfficientNet, uses a compound scaling strategy to equally scale the resolution, depth, and breadth of all backbone, feature network, and box/class prediction networks at the same time ensuring maximum accuracy and efficiency while working with limited computational resources.

Most detectors use multiple basic boxes, or anchors, which are bounding boxes defined by multiple boxes with different aspect ratios. A good regression of projected bounding boxes depends on the design and number of anchors being used to generalize the location, size, aspect ratio, and classification of training data. Each spatial cell in the output feature map predicts several boxes. Each box prediction is encoded as x- and y- offsets relative to the cell center, and width- and height-offsets relative to the corresponding anchor [71]. There are also anchorless object detection architectures such as CenterNet [72].

Robustness to small image translations is a highly desirable property for object detectors [73]. However, recent works have shown that CNN-based classifiers are not shift invariant [26]. Modern object detection architectures, no matter if one-stage or two-stage, anchor-based or anchor-free, are sensitive to even a one-pixel shift to the input images [73]. When several possible solutions to this problem are investigated it is seen that none of these methods can provide full shift-equivariance.

2.4 Generative Adversarial Networks (GANs)

Generative adversarial networks (GANs) for image synthesis mainly use deep convolutional network architectures and consist of a generator and a discriminator. GANs have two models which are trained simultaneously: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability of whether a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D, a unique solution exists, with G recovering the training data distribution and D equal to 1/2 everywhere. In the case where G and D are defined by multilayer perceptrons, the entire system can be trained with backpropagation [74].

Unlike the previous loss function of deep convolutional networks, the GAN loss function includes the image's perceptual loss, which comprises content loss and generation loss. The disparity between the real image and the artificial image created by the generation network gives generation loss.

There has been significant interest in GANs in the recently [75, 74, 76, 77] and they have been applied to various domains including computer vision [78, 79, 80], natural language processing [81, 82], time series synthesis [83, 84, 85], semantic segmentation [86, 87, 88]. Also, GANs have a wide range of application areas such as generating samples for image datasets, photographs of human faces, realistic photographs and cartoon characters, image-to-image translation, text-to-image translation, semantic-image-to-photo translation, face frontal view generation, generating new human poses photograph editing, face aging, photo blending, super-resolution, photo inpainting, clothing translation, video prediction, 3D object generation. Using generative adversarial networks (GANs) is promising for enhancing images and removing artifacts [89, 90, 26].

Despite their considerable achievements, there are a number of issues of GANs: mainly instability in training, difficulties in the generation of high-quality images, and improving image diversity [91].

In this study, we adapt the cycle spinning (CS) method and integrate it into a GAN for this purpose.

2.4.1 Image enhancement using GANs

It has been widely acknowledged that unpredictable impairments such as illumination, noise, and severe weather conditions (i.e. rain, snow, and fog) adversely affect the performance of many computer vision algorithms for object detection, classification, and tracking [26]. Recent years have witnessed significant progress in single image de-raining. The progress in this field can be attributed to various prior studies [92, 93, 94, 95, 96] and deep convolutional neural network (CNN)-based models [97, 98, 99, 4, 26]. GAN can be used for evaluating and analyzing performance differences between synthetic rainy images and real-world images by defining the strengths and limits of each method [100].

There are early methods based on multiple frames based on de-raining algorithms and using video by taking the average intensity of the detected rain streaks from the previous and subsequent frames. However, those methods are not applicable to single image de-raining [101, 101, 102, 103, 104, 105, 106].

Cleaned images used some other de-raining algorithms are beneficial but they tend to have unsatisfactory performances on real images with complicated scenes and rain forms remove [92, 93, 94, 95, 96].

Convolutional Neural Network (CNN) based algorithms have achieved success for single image de-raining. Progress in single image de-raining can be attributed to various natural image priors [101, 102, 107, 104, 92, 105, 106] and deep convolutional neural network (CNN)-based models mentioned before. Rain models in the literature can be investigated in three major categories: rain streak, raindrop, rain, and mist. One of the state-of-the-art models is the DeRaindrop algorithm [4] which is a CNN-based single-image raindrop algorithm. It aims to improve the images by cleaning the raindrops on a glass window or camera lens. As with all rain removal algorithms, the type and size of the rain stain are unknown and information is lost on the rainy region

of the image. To resolve the problem, an attentive generative network using adversarial training is used and the DeRaindrop algorithm has been applied. The DeRaindrop algorithm uses an attentive generative network to generate raindrops mask. Then it focuses on the rain regions and removes background interference.

As a result, the DeRaindrop algorithm is a successful method for raindrop images according to objective experimental results, recently. The CS method can be used for different aims such as dehazing [108, 100, 109], derainstreak [110]. Because such fluctuations adversely impact vision systems that rely on small image features for tracking, object detection, and recognition and there is no single best-deraining algorithm for all rain types.

CHAPTER 3

METHODOLOGY

3.1 Cycle-spinning (CS) method

Eq. 1 summarizes the CS method in wavelet domain, where $S(i,j)$ corresponds to 2-D circular shift, W is the wavelet transform, k_1, k_2 are the maximum number of shifts and T is a thresholding operator [32]:

$$\hat{y} = \frac{1}{k_1 k_2} \cdot \sum_{i=1,j=1}^{k_1 k_2} S_{-i,-j} \left(W^{-1} \left(T \left(W \left(S_{i,j}(y) \right) \right) \right) \right) \tag{1}$$

The fact that various coefficients may be canceled out by different iterations for the same shift value i and each iteration's denoising procedures will not always correlate to the same projection complicates the method. Despite these difficulties, the CS method has been proven to converge [33]. The proof mentioned here does not imply that the CS method estimates converge to the true signal, it shows that the CS method converges to the intersection of a series of spaces containing the true signal, and therefore the final estimate is close to the true signal.

In these studies, the CS method is proposed as a new method for image processing to reduce ringing and impairments commonly associated with conventional resolution enhancement approaches due to the unavailability of higher spatial frequencies. The following support that the CS method is an effective way of image resolution enhancement [111, 112, 113, 114].

During the acquisition process and down-sampling of a digital picture, there is some degree of high-frequency information loss. The resolution enhancement problem is

defined as estimating the missing higher spatial frequency information using the low-frequency information available. Based on this information, the CS method has been shown to restore high frequencies. The main idea of the CS method is based upon using non-shift invariance. When applied to image processing this results in different consequences. In fact, shifting the image in image processing generally results in undesired consequences and usually distorts it [115, 55, 7]. However, if non-shifted invariance is employed, such distortion may be eliminated. This has been shown in the resolution enhancement of the images by [1]. The recovery of high-frequency data entails the recovery of edges as well. Edges are the most crucial characteristics of an image since they transmit substantial information about it [38]. In [38] study, the unknown high-resolution (HR) image is generated using wavelet-domain zero padding (WZP) as a new approximation to the CS method. First, a number of low-resolution (LR) images are generated from spatial shifting, then wavelet transforming is applied and discarding the high frequency (HF) coefficients. Second, WZP processing is applied to all images. Finally, these intermediate HR images are re-aligned and averaged to give the final HR reconstructed image. As a result, visual artifacts and oscillations around signal discontinuities, called pseudo-Gibbs phenomena, are fixed by the shift variant property of the transform. We aim to apply the CS method, whose effectiveness has already been shown in wavelet domain processes, to deep learning algorithms for image processing.

In the light of all this information, we will be discussing how the CS method can be applied in the field of deep learning. Deep convolutional neural networks (CNNs) extract local features and learn spatial representations via convolutions in the spatial domain. The objects in most of the images have consistent relative pixel densities, which convolutions may take advantage of [116]. Pixel density is a calculation that returns the number of physical pixels per inch on an image as important as the resolution. For dense pixel prediction tasks, the loss is spatially varying because of varying scene elements on a pixel grid. For this reason, we proposed a new approach influenced by the CS method and adapted it to deep learning algorithms. In this approach, the basic operation is the convolution operation instead of wavelet transformation. Convolution is a fundamental image processing and computer vision operation, as

well as a key component of CNN architectures [117]. When, as prior information, it is known that a function of a limited number of surrounding pixels is helpful, it may utilized to exploit local information. For example, convolutions in the first layer of a convolutional network may be detecting the edges using local information when analyzing images. Because the statistics are expected to be similar at different locations of the image, same edges may occur at various locations in the image. This leads to the idea of sharing parameters across the network [118]. Listed below are the major parts covered in this dissertation to explore the adaptability of the CS method to the field of deep learning:

- Integrating CS into GAN-based models,
- Integrating CS into the convolutional layer of CNN-based models.

3.1.1 Integrating CS into GAN-based models

As the first step, we adapted the CS method to a GAN-based de-raindrop algorithm. GANs are composed of three main parts: the generator, the discriminator, and the loss function. In order to create a good generator G to fool the learned discriminator D and to make the discriminator D good enough to distinguish the image from real ground truth, the method updates G and D by using a loss function [119]. It is important to use all information in an image during the convolution process in order to properly employ the CS method, therefore, CS has been employed to wrap around the whole algorithm. This is different from its other applications in this thesis, in which it only wraps around a convolutional layer. The process is illustrated in Figure 5.

3.1.2 Integrating CS into the convolutional layer of CNN-based models

The second step is to adapt the CS method to use with CNN-based classification and object detection models. Classification involves the prediction of the class to which an image belongs. Some classifiers are binary, resulting in a yes/no decision. Others

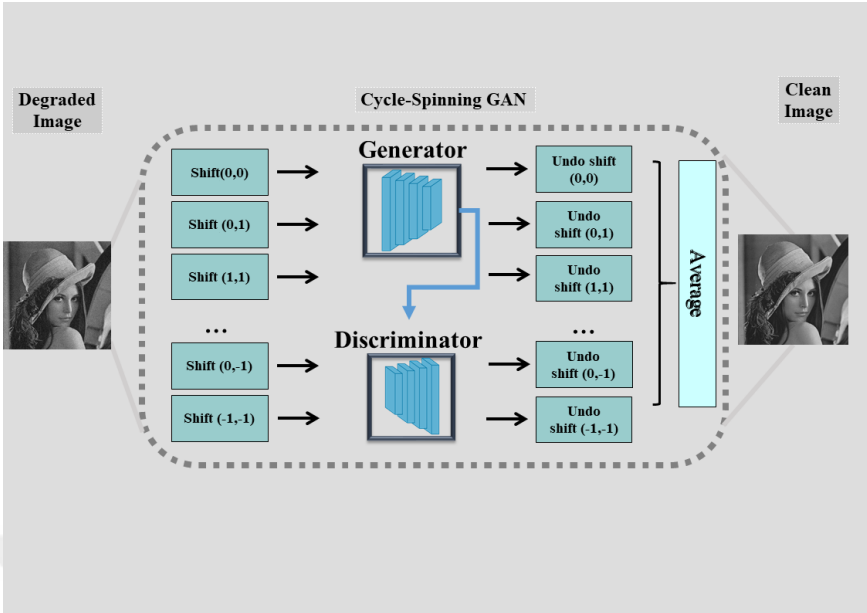


Figure 5: The CS block diagram illustrates the adaptation of the CS method by wrapping the algorithm.

are multi-class, able to categorize an item into one of several categories. Classification is a very common use case of machine learning—classification algorithms are used to solve problems like image recognition and object detection. So, in addition to classification, we also apply the CS method to an object detection algorithm. In regular applications, the CS method wraps around the wavelet transformation, to adapt it to use in classification and object detection networks, CS wraps around the first convolution operation. In particular, the first convolution operation is chosen because, in signal processing applications, low-pass filters are often placed before downsampling to avoid aliasing [7]. The CS method can also be added to the convolution operations on different layers of the algorithm. The illustration of the CS method adaptation to classification and object detection algorithms is shown in Figure 6 and the details of the deep cycle-spinning convolution are shown in Figure 7.

If we denote the proposed CS method according to the spatial domain in Eq. 2, S(i,j) means 2-D circular shift, C means a convolution, k_1 k_2 is the maximum number of shifts:

$$\hat{y} = \frac{1}{k_1 k_2} \cdot \sum_{i=1,j=1}^{k_1 k_2} S_{-i,-j}\left(C\left(S_{i,j}(y)\right)\right) \tag{2}$$

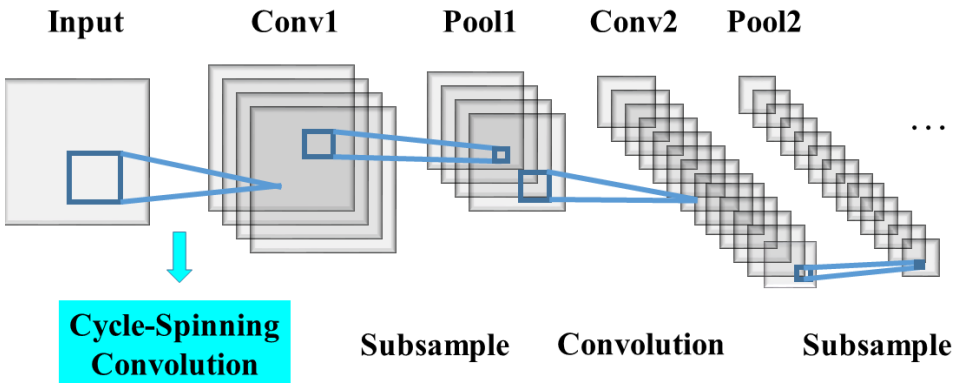


Figure 6: Illustration of the proposed algorithm in which cycle spins by convolving the input via basic convolution at each iteration.

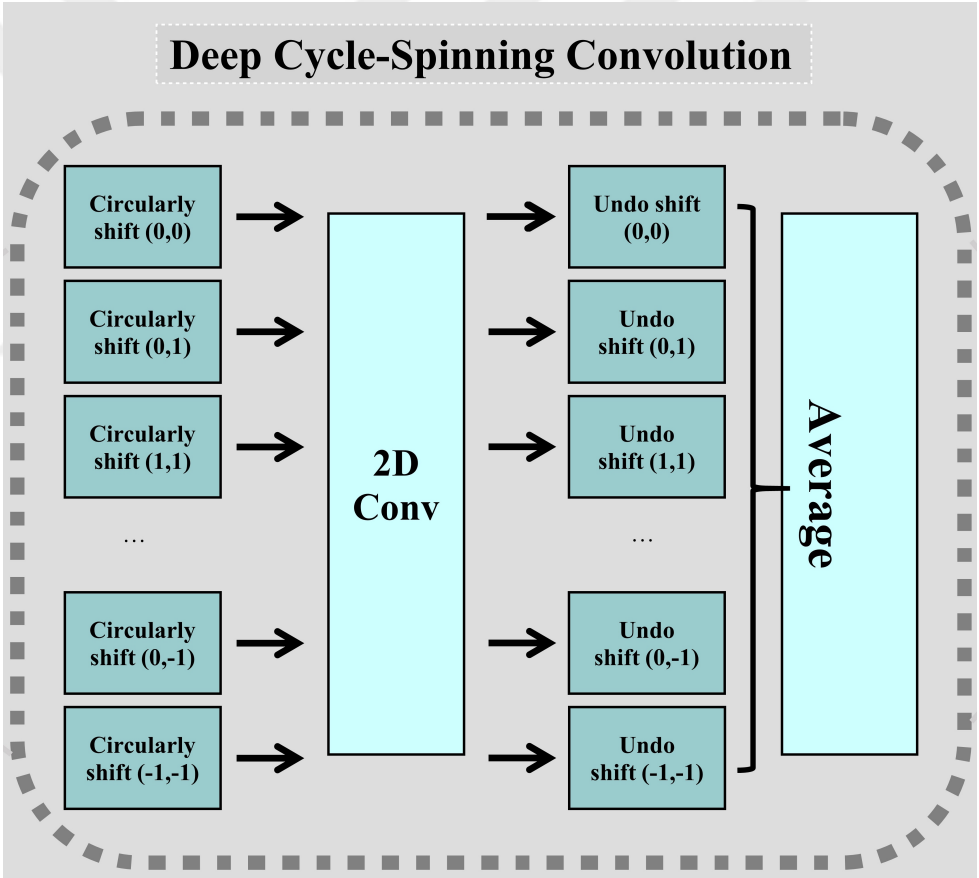


Figure 7: Flow of the cycle-spinning convolution method.

Firstly, in Figure 8, it is shown how different values are calculated at the end of the pooling. It can be seen more clearly in the visualization that the weights for "to the right of the center, down by one" and "to the left of the center, up by one" is differ-

ent, therefore if the output-centered filter is simply taken and used in a convolution framework, it will yield different weights as a result of pooling [120]. Because of this reason, models cannot utilize information effectively because it only adopts a fixed stride strategy, which may result in poor generalization ability and information loss, especially for edges and small details of objects. However, the CS method's final estimate is obtained by simply linearly averaging the shifted versions' estimates. The errors in the estimates are not completely dependent and sequentially averaging them results in noise reduction, anti-aliasing, and edge-preserving.

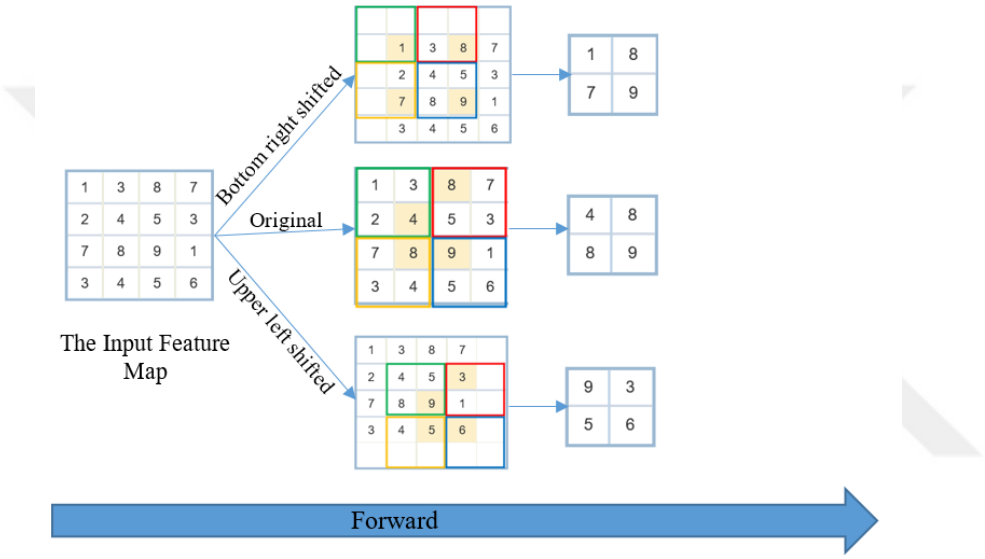


Figure 8: Output channels of a standard convolution operation on input and element-wise multiplication of matrices followed by a max-pooling are shown for the illustration.

Secondly, to show the behavior of the CS method two functions such as f and g that act on the same domain of inputs can be assumed. When two functions are convolved, the convolution operates like a function in the sense that we can compute the value of the convolution at a certain moment t. Calculating a convolution between f and g, at point t, it is taken the integral over all values α between negative and positive infinity, and, at each point, multiplies the value of $f(x)$ at position α by the value of $g(x)$ at t.

We literally take one signal and shift it in the time domain. The integral of the overlapped section is then calculated by multiplying this signal with the other signal. As a sample, we convolve the $f(t) = e^{-|t|}$ function with a sinc-squared function. Results

of convolution with and without the CS method are illustrated in Figure 9 and Figure 10 which shows what we get when the CS method has been added to the convolution of the $f(t)$ and $g(t)$ functions: smoothness and information aggregation.

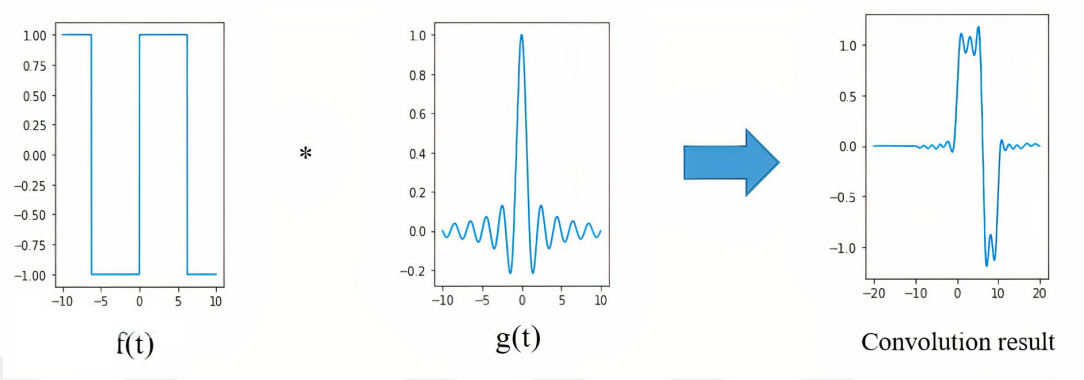


Figure 9: Convolution results of f(t), g(t) functions without the CS method employed.

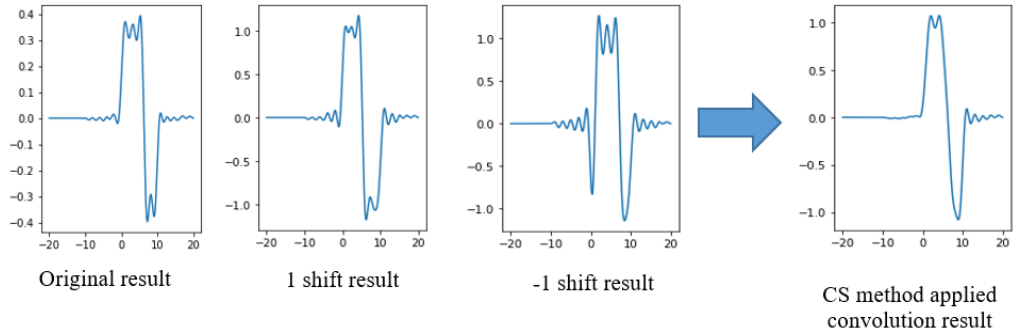


Figure 10: Convolution results of f(t), g(t) functions after the employment of the CS method.

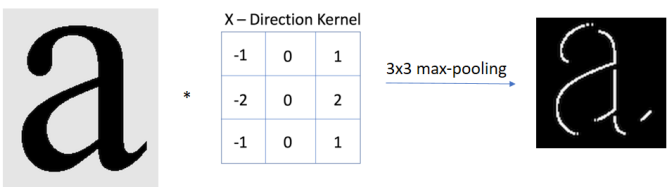


Figure 11: The basic model for analyzing the results of 2D image processes.

To give an idea and to justify the usage of the CS method, we have analyzed 2D image examples. Natural images have varying texture characteristics and frequency contents. Therefore, test patterns are used for justification. A convolution operation

as seen in Figure 11 using Sobel x direction kernel ([[-1, 0, 1], [-2, 0, 2], [-1, 0, 1]]) and afterward a max-pooling operation is used for test pattern experiments.

Firstly, we have chosen a well-known black-and-white TV test pattern. The selected image has detailed vertical lines which may be affected in different ways with the application of downsampling, strided convolution, and the like. [26]. When downsampling a signal, it is a general practice to use low-pass filtering for anti-aliasing [24]. As the averaging process done by the CS method resembles a general low-pass filter, we sought their differences in this test pattern experiment. Low-pass filters are used to smooth out harsh intensity changes. Noise reduction is one apparent use of smoothing; as random noise generally has rapid intensity shifts. In the second row of Figure 12, it is observed that regular convolution causes aliasing and the details of vertical lines are lost. This effect is more pronounced at the higher frequencies towards the right side of the image. The CS convolution results (third row), exhibit less aliasing due to the averaging-out effect and the lines are more distinctly visible. In this case, more information is preserved and propagated forward to the subsequent layers for further processing. 3×3 low-pass filtering after the convolution operation has an anti-aliasing effect (last row).

On the other hand, significant amounts of detail are lost in this instance due to filtering, especially at the higher frequencies. The results confirm that the CS convolution operation provides a good balance between regular convolution and convolution followed by low-pass filtering, as it prevents anti-aliasing while preserving the details.

The same process was applied to test images consisting of a square and the letter "a" (Figure 12). It is observed that the horizontal lines and the right side of the vertical line disappear as a result of the max-pooling applied after convolution. The CS convolution version preserves the horizontal lines to a degree and the vertical line is more visible. On the other hand, horizontal lines and vertical lines become blurry when low-pass filtering is used. Analysis of the results for the letter "a" shows that more information is preserved around the upper curve of the letter “a” for the CS method applied version, just as in the square and round object examples. Also, the anti-aliasing effect is seen in the overall shape of the “a” letter. As in the other cases

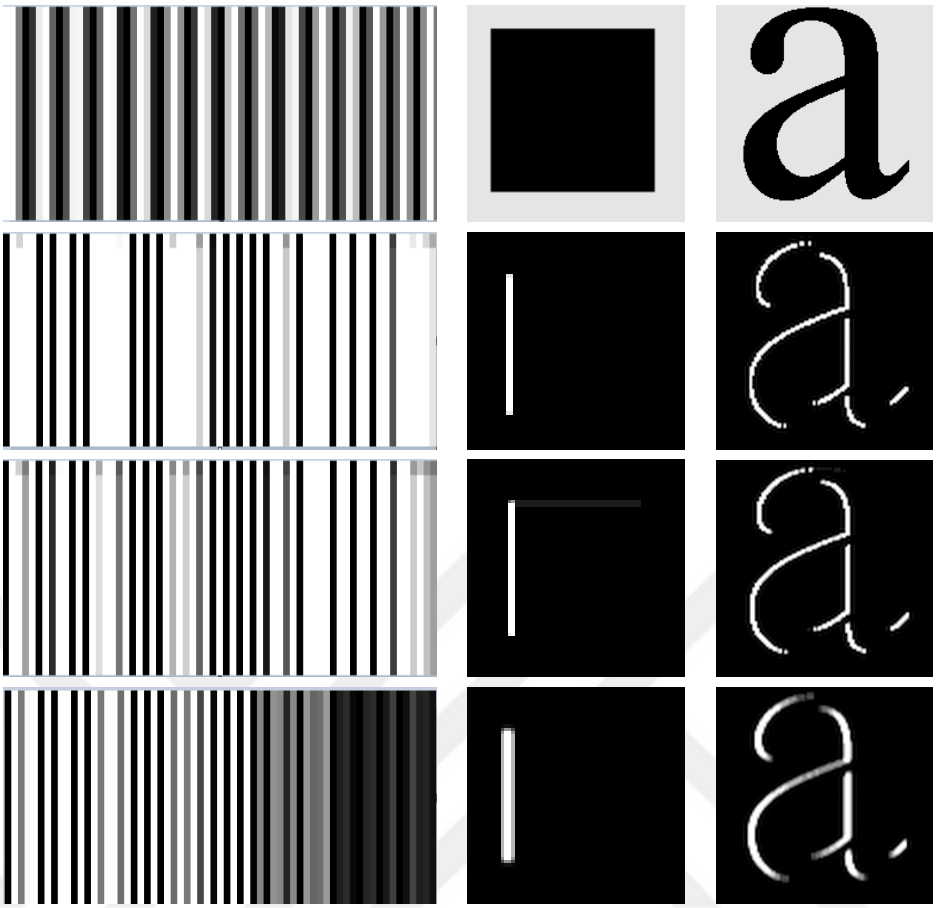


Figure 12: Visual comparison of pooling results on synthetic test patterns. Original patterns are shown at the top row, followed by max-pooling results using regular convolution, using CS integrated convolution, and using a low-pass filter before convolution.

above, low-pass filtering results in blurring and there is more data loss around the upper curve of the letter “a”.

To analyze the effects of the CS method in the presence of noise, we repeated the experiment using the same algorithm by adding Gaussian noise to the Lena image seen in Figure 13.

In the first row of Figure 14, the results of the convolution, CS applied and low-pass filtered versions of the Lena image are shown sequentially. In the second row of Figure 14, 8 times magnification of Lena’s hat section is shown for a better view of the details. As a result, it is seen that the CS method implementation removes

Figure 13: Original image of Lena (left) and noisy version (right) with noise.

noise without losing details, as in the test pattern images. When the results have been compared with the low-pass filtered version, it is also seen that the edge lines of the objects are not blurred in the CS-implemented version. As is seen in the sample experiments, the healing effects of the CS method are the motivation behind this research.

These results on toy examples reveal that the CS method added versions of convolution preserve the edges of the objects in the image and have a noise-suppressing effect. In a multi-layer architecture, this implies that more information is passed onto the subsequent layer. Next, we focus on the question 'what gains will be achieved in the deep learning algorithms?'. The research paper titled "Axiomatic Attribution for Deep Networks" defines axioms for the correctness of attribution methods that generate attribution scores for inputs to deep networks [121]. That study highlights two such axioms as sensitivity and implementation invariance. These properties are good for understanding the background of the CS method in deep learning. There are several methods to calculate and visualize the results. We focus on the attribution method which is called the Integrated Gradients. The advantage of Integrated Gradients is that it does not require network instrumentation and can be easily computed with just a few calls to the gradient operation.

An attribution method scores the input data based on the predictions the model makes, i.e. it attributes the predictions to its input signals or features, using scores for each feature. For example, a score is generated for each feature in the input image the model predicted for and the scores associated with each of these features give an in-

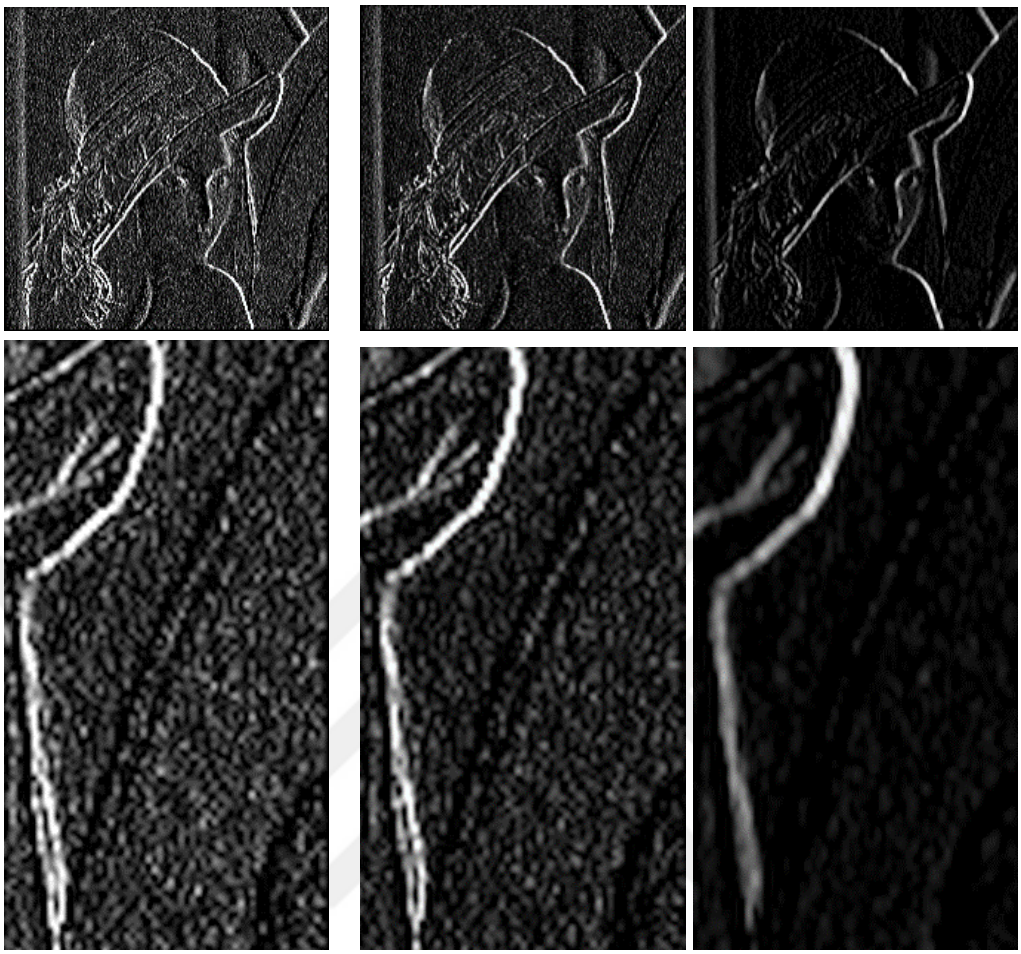


Figure 14: Visual comparison of pooling results on noisy Lena image. Detail of hat ($8\times$) images are shown in the second row. The max-pooling results using regular convolution, using CS integrated convolution, and using low-pass filter before convolution is in the top row.

dication of their role in prediction. The gradient is the signal that tells the neural network how much to increase or decrease a certain weight/coefficient in the network during the backpropagation. It relies heavily on the input features for this task, therefore, the gradient associated with each input feature with respect to the output can help us get a clue about how important a feature is.

We used a basic network based on the architecture proposed in the CIFAR-10 beginner tutorial to understand the effect of the CS method. The general architecture application. This model has trained only 8 epochs for classes = ('plane', 'car', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck') using Captum library [122] which

interprets output predictions with respect to inputs. We have trained with and without the CS method and showed visualization results in Figure 15.

The CS method has been applied to the first convolution part of the model. Thus, each batch is trained with 1 and 2-shift CS methods. Our experiments show that the CS_{1+2} versions of the CS method are more successful than expected. Because of this reason, we have employed both 1 and 2 shifts CS in this model. The CS method's final estimate is obtained by simply linearly averaging the shifted versions' estimates. The errors in the estimates are not completely dependent and sequentially averaging them yields a noise reduction. It is crucial to achieving the same size results from each phase of the CS method while averaging.

In the model trained on the CIFAR-10 data, we have examined the deep neural network model with the aim to understand which of the features was most important and how the network reached its prediction. Our selection is neuron (importance) attribution algorithms because we search not only for classification but also for object detection. In the first experiment, we applied the CS method on the first convolution which has the following parameters: channel=3, height =6, kernel size= (5, 5), stride= (1, 1). We also experimented with adding the CS method on the second convolution of the model which has the following parameters: channel=6, height=16, kernel size= (5, 5), stride= (1, 1). When the visualization results in Figure 15 are inspected, it can be seen that the correlation with the data is higher with the addition of the CS method and the contours of the objects in the image became more pronounced.

To provide some insight into how the proposed model comes to a decision and how it compares against the baseline, we used Gradient-weighted Class Activation Mapping (Grad-CAM) visualizations. The Grad-CAM uses the gradients and highlights the important regions in the image [3]. This technique creates heatmaps using a trained neural network after training is finished and the parameters are fixed. It is also known as post-hoc attention. This is different from trainable attention, which entails learning specific parameters during training to build attention maps.

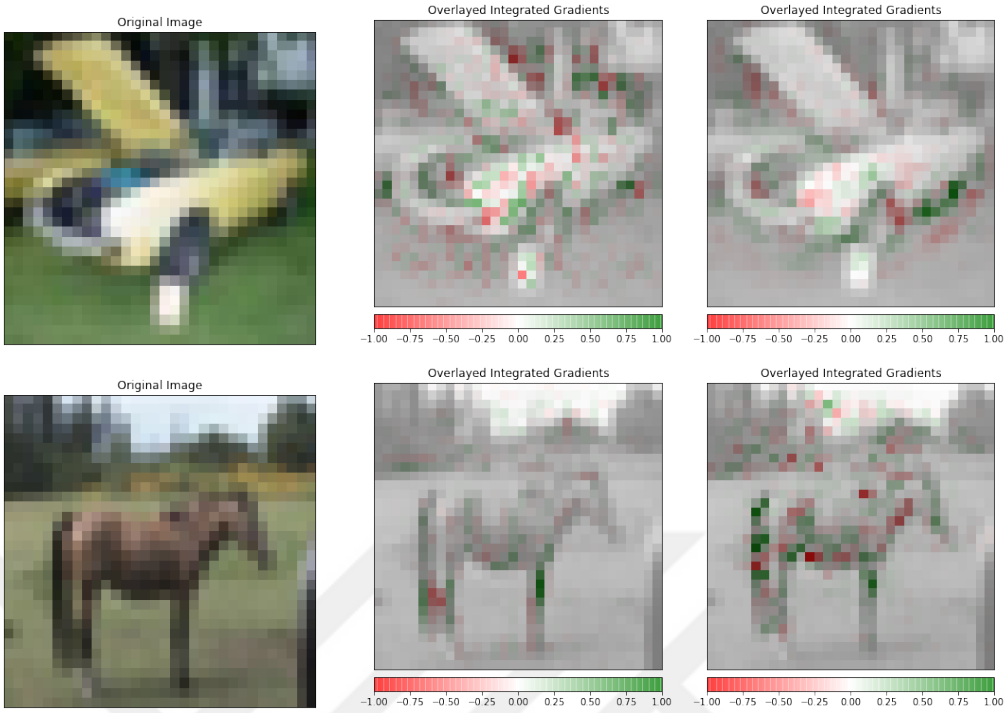


Figure 15: Integrated gradient visualization results of the model on CIFAR-10 test images. Left-most column: original image; middle column: its corresponding regular method result; right-most column: CS_1 method adaptation result.

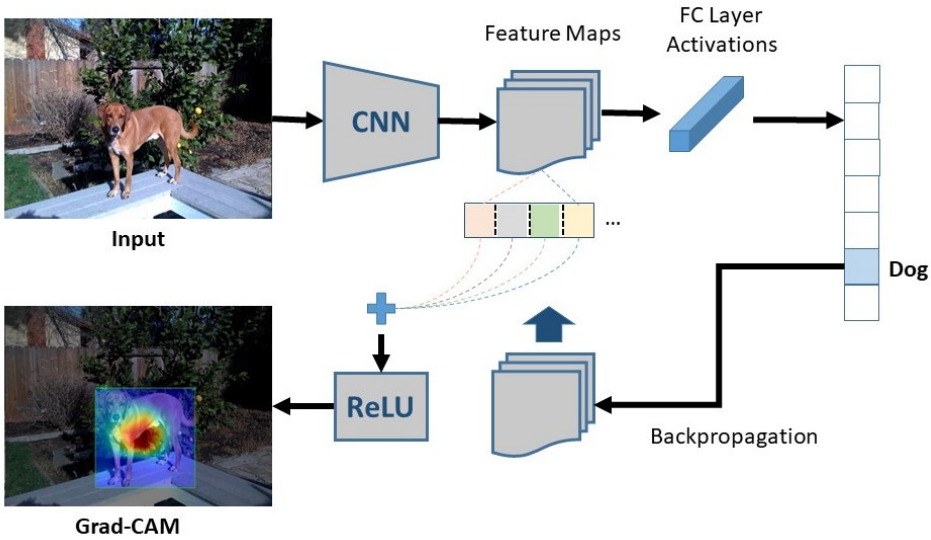


Figure 16: Overview of Grad-CAM heatmap. (Adapted from [3])

A feature map shows the location of features in the image. As shown in Figure 16, each channel in the feature map array is multiplied by the gradient of the highest predicted class for the input image. The gradient is relative to the activations of the last convolution layer. By doing this, the magnitudes of the fields that help with the prediction increase. Then all channels are summed to get the class activation heatmap. This heatmap can be used to analyze where the CS method is paying attention while predicting the given image and compare it against the attention of the original model.

As seen in Figure 17, the proposed method concentrates more on the areas which have more distinctive information about the object (i.e., focusing more on information extracted toward the face and eye of the cat in this specific instance). The red and yellow areas indicate the locations to which the model pays attention while making predictions. Blue areas are the locations where activation is lower. For CS_1 and CS_{1+2} results, the heatmap is observed to move towards the face of the cat, indicating that CS enables the model to be a better learner.

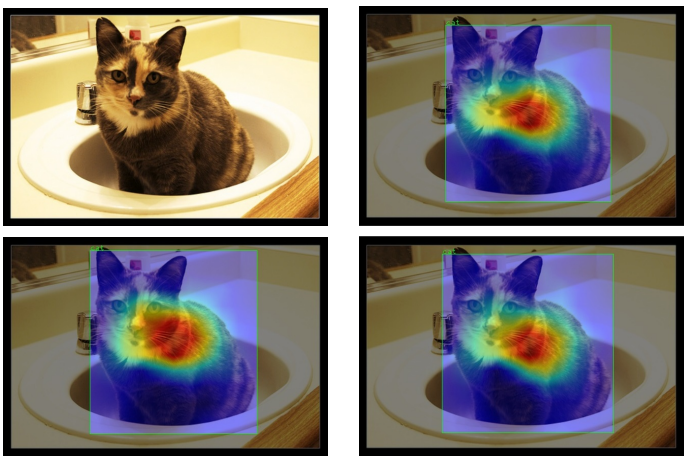


Figure 17: Comparison of the GradCAM visualizations of the baseline, CS_1, and CS_{1+2} applied versions of the object detection algorithm. Original image (top left), baseline algorithm result (top right), CS_1 applied version (bottom left), and CS_{1+2} applied version (bottom right).

3.2 Shifting

Shifting an image means moving all the pixels of the image while keeping the image dimensions the same. This means that some pixels will be clipped from the image and there will be a region in the image where new pixel values must be specified.

CS method involves spatial shifting of pixels of an image. There are alternative shifting approaches, some of which may result in the loss of some information about the object in the image, as shown in Figure 18. This may reduce the performance of detection objects which are not fully in the frame or at the edge of the image like in Figure 19.

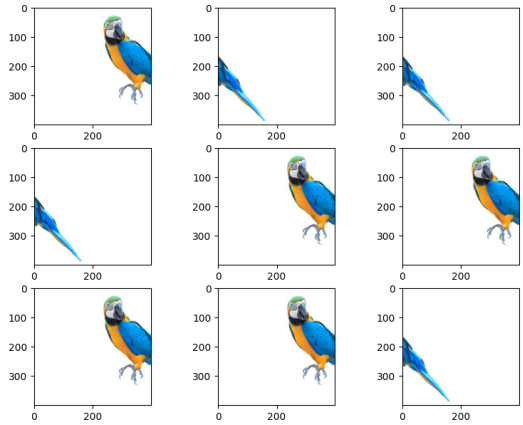


Figure 18: Shifting examples of an image (Generated by Keras ImageDataGenerator Class).

To overcome this problem, we decided to shift images using padding. In the context of CNNs, padding refers to the pixels added to the image borders before it is processed by the kernel of a CNN. For example, when zero-padding is used, then every pixel value that is added will be of value zero. Padding works by extending the area that a convolutional neural network processes.

In our experiments, we have evaluated constant, reflect, constant, replicate, and circular padding, as shown in Figure 20. It is shown that zero padding drives CNNs to encode position information in their internal representations in contrast to conventional padding types such as reflection, replicate, and circular [124]. According to

(a) Room scene

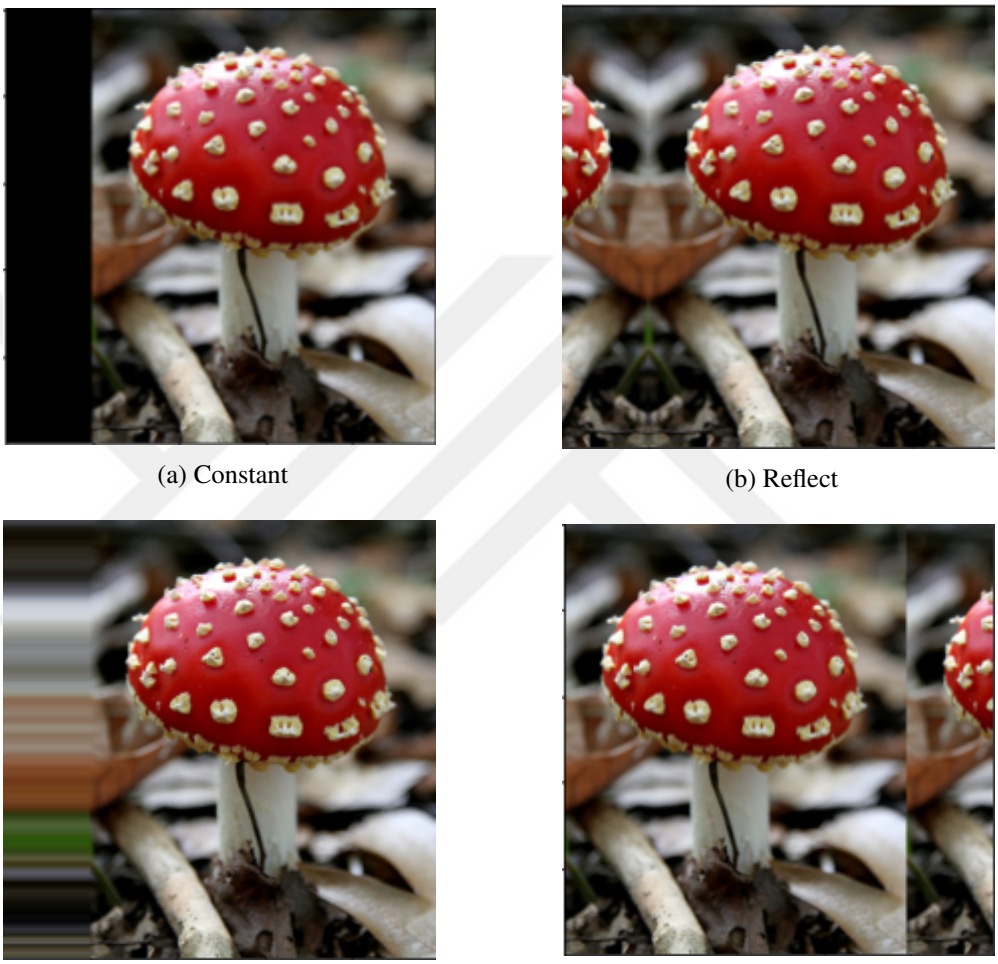
(b) Work scene

(c) Group photo

(d) Desk objects

Figure 19: Examples of images with border objects that are not fully in the frame, selected from the MS COCO 2014 image set [123]. The images show different types of images sampled in the dataset, including objects, and scenes.

Alsallakh et.al. circular padding is the second capable padding type in CNNs while reflect and replicate padding is not as beneficial at the image boundaries. In circular padding, When the convolutional kernel hits the edge, it rolls to the other side, and similarly, when shifting, pixels are rolled off from one edge to the other. Information is lost on one side of the shift and must be filled in on the other. Especially for small-sized datasets, circular padding is recommended [26]. As well, [29] gets slightly better performance in experiments in feature space.



(a) Constant

(b) Reflect

(c) Replicate

(d) Circular

Figure 20: The representation of four different padding methods on a sample image from the CIFAR-10 dataset.

CHAPTER 4

INTEGRATING CS INTO GAN-BASED MODELS FOR RAINDROP REMOVAL

Based on the literature survey in Section 2.3.1, we decided to adapt the CS method into Attentive GAN [4]. DeRaindrop (using Attentive GAN) is a method developed specifically for raindrop removal and it is reported to be the best-performing method for this particular problem in a recent benchmarking paper [125]. In this work, we use De-Raindrop as a basis due to its proven performance and adapt the CS method to further improve its state-of-the-art results.

4.1 DeRaindrop algorithm

The deRaindrop algorithm uses an attentive generative network to generate a raindrop mask and then remove the raindrops by using it. Attentively focusing on rain regions makes the algorithm stand out.

4.1.1 Datasets

Similar to most of the currently used deep learning methods, our method also requires a relatively large amount of data with ground truths for training. Since a dataset for raindrops attached to a glass window or lens does not exist, a dataset using two pieces of the same glass: one sprayed with water, and the other clean, 1119 pairs of images, with various background scenes and raindrops, were created with Sony A6000 and Canon EOS 60 cameras for image acquisition by Qian et al. ([4]). To create a variety

of raindrop photos and limit the reflecting impact of the glass, a 3 mm glass slab was utilized with a distance ranging from 2 to 5 cm between the glass and the camera.

Training has been done using the data set presented in ([4]) which consists of 720×480 pixels, 860 real images, and 860 images having raindrops. Evaluations have been done on 58 test images having a resolution of 720×480 pixels. All training and testing processes have been carried out on Nvidia GeForce GTX 1080 GPU. All the quality metrics have been calculated using MATLAB R2017b. Code is publicly available at `https://github.com/UlkuUZUN/cs-attentiveGAN`.

4.1.2 Architecture

As shown in Figure 21, the algorithm architecture is generic and any artefact removal algorithm having a shift-variant nature could be applied to it. In this network, there are two main parts such as generator and discriminator networks, as in the known GAN definition [74]. The attentive generative network is made up of two sub-networks: an attentive-recurrent network and a contextual autoencoder. The attentive recurrent network's goal is to identify regions in the input image that require attention. These are primarily the raindrop areas and their surrounding structures. The contextual autoencoder works in order to provide better local image restoration. The discriminative network must focus on the evaluation. In order to localize and capture the characteristics of the targeted regions in an image, visual attention models have been used by writers. Visual attention is very important for producing raindrop-free background images since it tells the network where to focus removal efforts. Each block in the recurrent network consists of five ResNet [126] layers. These residual layers feed into the next layer and directly into the layers about 2 hops away and that aid in the extraction of features from the input image lost in previous layers. In this network, also a convolutional LSTM unit [127], and convolutional layers are used for producing the 2D attention maps. In this network, attention maps are composed of values between 0 and 1, and these values have meaningful consequences in the presence of raindrops.

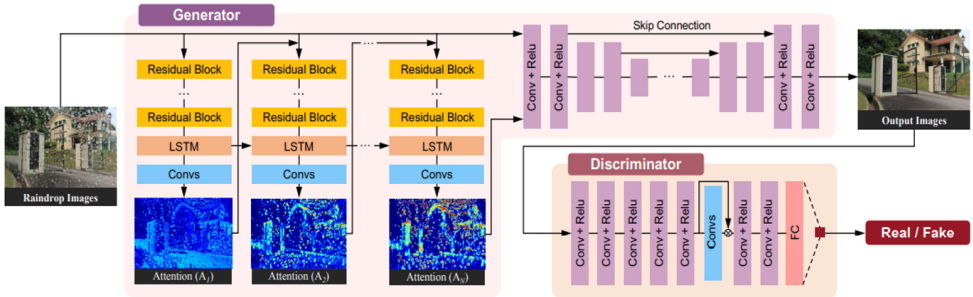


Figure 21: Architecture of the attentive generative adversarial network [4].

Two different variants of the proposed method have been evaluated. The first one integrated the CS method into the test phase only. In this case, the algorithm is trained without any modifications. The CS method is only applied in the test phase by using shifted versions of the input image, then undoing the shift at the output, and taking an average of both to reach the final result. The CS method is applied in the training phase in the second case. In the application of the CS method with 1-pixel shifts, in addition to the original image, combinations of 1-pixel shifts in x and y are used: [(1, 1), (1, 0), (0, 1), (-1, -1), (0, -1), (-1, 0), (-1, 1), (1, -1)]. The 1 and 2-pixel shift version uses the original image, combinations of 1-pixel shifts, as well as combinations of 2-pixel shifts : [(1, 1), (1, 0), (0, 1), (-1, -1), (0, -1), (-1, 0), (-1, 1), (1, -1), (2, 2), (2, 0), (0, 2), (-2, -2), (0, -2), (-2, 0), (-2, 2), (2, -2)].

4.1.3 Evaluation metrics

Processing of an image can result in degradation of image quality. Image quality evaluation methods are classified as objective or subjective. The distinction between the two approaches is that one focuses on the properties of signal processing in various imaging systems, whereas the second focuses on the perceptual factors that make an image appealing to human viewers. The performance of raindrop removal has been examined using both quality approaches in our Cycle Spinning Generative Adversarial Network experiment. All the quality metrics have been calculated using MATLAB R2017b.

Firstly, we have calculated Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) ([128]). The PSNR equation (Eq. 3) computes the peak signal-to-noise ratio between two images in decibels using Mean Square Error (MSE). The MSE (Eq. 4) is the cumulative squared error between the processed and the original image. The better the quality, the higher the PSNR.

The PSNR is defined based on the ratio of the square of the maximum possible pixel value of the image (R) and MSE:

$$PSNR = 10 \log_{10} \left(\frac{R^2}{MSE} \right) \tag{3}$$

MSE is defined as the difference between a noise-free $m \times n$ image I and its processed version K:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2 \tag{4}$$

SSIM is a well-known quality metric used to measure the similarity between two images. Instead of using traditional error summation methods, it is based on modeling any image distortion as a combination of three factors that are loss of correlation, luminance distortion, and contrast distortion. The better the quality, the higher the SSIM.

Further, to evaluate the perceptual quality of restoration, three no-reference Image Quality Assessment (IQA) metrics have been used:

- Naturalness Image Quality Evaluator (NIQE) ([129]),
- Blind/Referenceless Image Spatial Quality Evaluator (BRISQUE) ([130]), and
- BLind Image Integrity Notator using discrete cosine transform (DCT) statistics (BLIINDS-II) ([131]).

The NIQE and BRISQUE algorithms employ natural scene statistics (NSS). The NSS is helpful for specifying an ideal observer's behavior in a natural task, generally by using signal detection theory, information theory, or estimation theory. NSS features

are used by different algorithms in different ways. For example, the NIQE algorithm extracts the NSS features from statistically significant blocks in the distorted image and then fits a multivariate Gaussian distribution to the image NSS features. Finally, the distance between the Gaussian distributions is used to calculate the quality score, as shown in Figure 22.

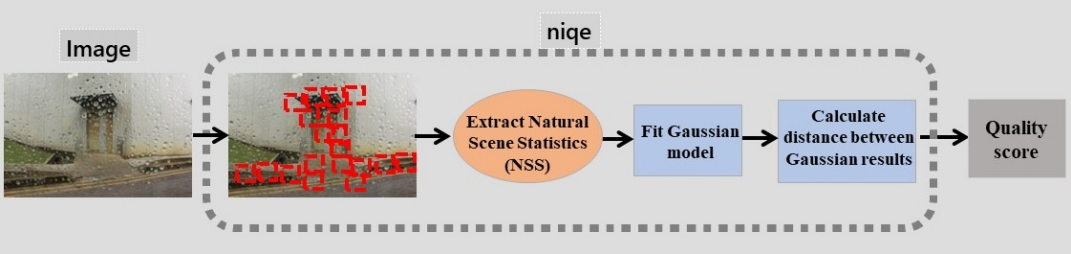


Figure 22: NIQE workflow diagram.

Also, the BRISQUE algorithm extracts the NSS features from the distorted image and predicts a quality score using support vector regression (SVR), as shown in Figure 23.

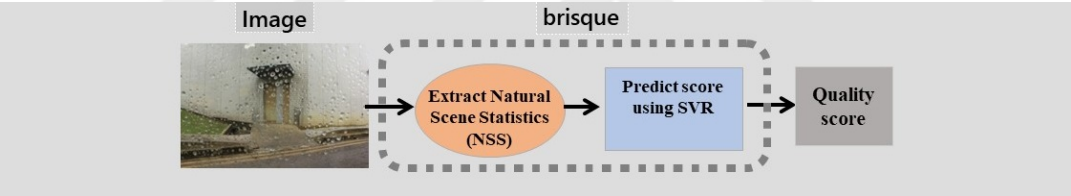


Figure 23: BRISQUE workflow diagram.

While NIQE compares the image to a default model computed from images of natural scenes, BRISQUE compares the image to a default model computed from images of natural scenes with similar distortions. Lower BRISQUE and NIQE scores indicate better perceptual quality. Because BRISQUE and NIQE results show how far from according to the viewers.

The BLIINDS-II relies on a Bayesian inference model to predict image quality scores given certain extracted features. The features are based on a natural scene statistic model of the image DCT coefficients. In ([131]) and ([132]), a smaller BLIINDS-II score indicates better perceptual quality, but to be consistent with full-reference metrics and compared with other studies [125, 100]. The bigger the value, the better the perceptual quality.

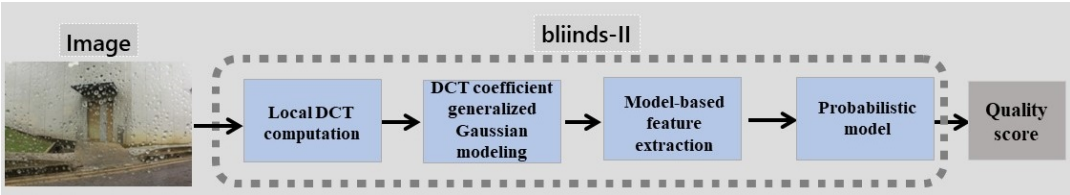


Figure 24: BLIINDS-II workflow diagram.

To test the object detection performance, we used the TensorFlow Object Detection API with the SSD MobileNet pre-trained model (on COCO) ([133]) and calculated the mean average precision (mAP) averaged over IOU thresholds in [0.3: 0.05: 0.99]. If the score of a detected object is higher than a threshold of 0.3, its label and bounding box coordinates are recorded and used in the performance evaluation.

4.1.4 Results

Table 1 summarizes the experimental results in terms of two full-reference metrics (PSNR, SSIM), three no-reference metrics (NIQE, BRISQUE, BLIINDS2), and a task-driven object detection metric (mAP).

Table 1: Average full- and no-reference evaluation results on raindrop added real-life images and object detection measure mAP results.

	PSNR	SSIM	NIQE	BRISQUE	BLIINDS-II	mAP
Original images	-	-	4.40	22.79	66.52	63.73%
Images with raindrops	24.19	0.89	**4.07**	23.82	**69.37**	60.40%
DeRaindrop	25.47	0.89	4.30	**17.88**	64.63	65.22%
1-shift CS (test)	26.09	0.90	4.44	18.26	66.39	68.68%
2-shift CS (test)	26.01	0.90	4.45	18.19	65.58	67.17%
2-shift CS (train)	**26.12**	**0.91**	4.29	18.40	66.20	**69.05%**

With regards to the full-reference metrics, images obtained with the baseline DeRaindrop algorithm ([4]) have higher PSNR results compared to the images with raindrops, i.e. average PSNR through the test set increases from 24.19 to 25.47. On the other hand, SSIM results exhibit no change and both have an SSIM value of 0.89. The proposed method has higher PSNR and SSIM results compared to DeRaindrop. In the one-shift testing only, PSNR increases to 26.09, and SSIM increases to 0.90.

When CS is applied during the training, both PSNR and SSIM values became higher; PSNR increases to 26.12 and SSIM to 0.91.

With regards to no-reference IQA metrics, it is interesting to see that the original images without raindrops have worse NIQE and BLIINDS-II scores than their counterpart images with raindrops. Application of CS does not result in any improvement compared to the original DeRaindrop algorithm in terms of NIQE and BRISQUE scores, while it has a better BLIINDS-II score. By their design, IQA metrics are dependent on the reference image databases. It is reported in the literature that the no-reference metric scores are not consistent with the subjective human-judgment evaluations [125]. These observations make use of no-reference metrics in assessing raindrop removal performance questionable.

(a) Ground truth image

(b) Corresponding image with raindrops

(c) Result obtained using DeRaindrop

(d) 1-shift cycle-spinning

Figure 25: Demonstration of a test image, raindrop applied version, and raindrop removal results. (a): The original door image. (b): The input image is degraded by raindrops. (c): The base algorithm result. (d): The CS method applied model results, where most raindrops are removed and structural details are restored.

Degradation of visibility inevitably affects the subsequent vision algorithms in intelligent transport and outdoor video surveillance systems [134]. The performance of object detection, which is an important element in these systems, deteriorates in the presence of raindrops and is thus largely affected by the quality of image enhancement. For this purpose, mAP, which is an indicator of object detection performance, has been used as a task-driven metric. According to the results, in line with the expectations, DeRaindrop algorithm results in an increase in object detection performance 65.22% vs 60.40%. The proposed method further increases the performance and the best results are obtained when it is applied at the training stage increasing the mAP up to 69.05%.

According to the experimental results, use of the CS method increases the performance of object detection and the image quality in terms of full-reference image assessment metrics (PSNR, SSIM). While no-reference metric results are inconclusive, it can be claimed that they are not suitable for use in this particular context due to the reasons described above.

Figures 25, 26 and 27 show some sample ground truth images, corresponding images with raindrops as well as results obtained with DeRaindrop and 1-shift cycle spinning applied at a test only. It can be observed that the images generated using the CS method have sharper characteristics compared to DeRaindrop results and exhibit fewer artefacts and blur.

Figure 28 shows the object detection results for some sample images. The object detection confidence scores are shown next to the bounding boxes. It can be seen that the confidence scores are lower when the images have raindrops and in some cases, the raindrops cause missed detections (for ex. missing the traffic light in the first row) or incorrect labeling (labeling the car incorrectly as broccoli in the third-row) of some objects. While DeRaindrop algorithm increases the detection confidence scores, the proposed method further increases these scores for many objects. A higher mAP score than the original images indicates the potential of the method for improving the object recognition performance for clean images and images with raindrops as well.

(a) Ground truth image

(b) Corresponding image with raindrops

(c) Result obtained using DeRaindrop

(d) 1-shift cycle-spinning

Figure 26: Demonstration of a natural scene test image, raindrop applied version, and raindrop removal results. (a): The original image. (b): The input image is degraded by raindrops. (c): The base algorithm result. (d): The CS method applied model results, where most raindrops are removed and structural details are restored.

To enhance the final algorithm performance, the CS method can be applied to both the training and testing phases of the network.

We demonstrated that the proposed method works successfully for GAN models on 1 and 2 shifts. Visual results and objective evaluation metrics confirm the success of the proposed method. Extension of the method to work with a higher number of conditions is trivial, however, this also increases the training time of the model.

The work presented in this chapter has led to a publication with the title "Cycle-Spinning GAN for Raindrop Removal from Images" at 16th IEEE International Conference on Advanced Video and Signal-Based Surveillance, AVSS 2019, Taipei, Taiwan, September 18-21, 2019.

(a) Ground truth image

(b) Corresponding image with raindrops

(c) Result obtained using DeRaindrop

(d) 1-shift cycle-spinning

Figure 27: Demonstration of a test image, raindrop applied version, and raindrop removal results. (a): The original natural scene image. (b): The input image is degraded by raindrops. (c): The base algorithm result. (d): The CS method applied model results, where most raindrops are removed and structural details are restored.

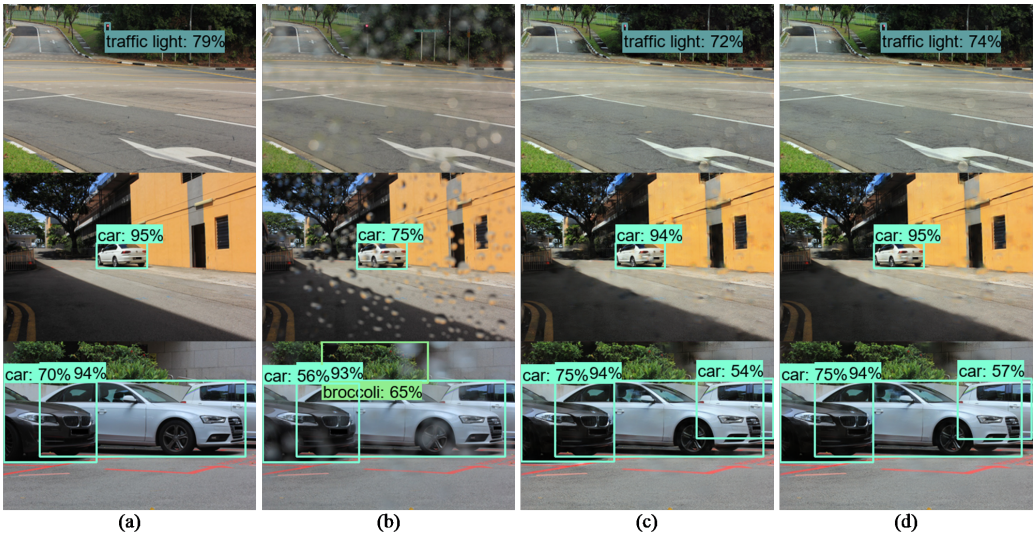


Figure 28: Object detection scores of (a) ground truth images, (b) images with raindrops, (c) DeRaindrop algorithm output, (d) 1-shift CS output.

CHAPTER 5

INTEGRATING CS INTO CNN-BASED MODELS FOR IMAGE CLASSIFICATION

CNN-based models have been successfully used in a wide range of classification applications. The main components of these models are, convolution layers, pooling layers, and fully connected layers. In addition, some layers have an activation function and a normalization step. One of the issues of modern CNNs is that they have no invariance to (small amounts of) translations, scale, or geometric distortion of the inputs [7, 8]. Because mainly used layers such as max-pooling, average pooling, and strided-convolution ignore the sampling theorem. A well-known solution against this is using an anti-aliasing mechanism. To enable anti-aliasing for CNNs, it has been proposed to include a content-aware anti-aliasing module before each downsampling operation in the network [7]. The mentioned method is a combination of two phases: the first phase is generating low-pass filters for different spatial locations and channel groups and the second phase is applying the generated filters to the input features. But applying the same filter to the whole image is problematic. This is because of the fact that other maps' spatial positions and channels would be different [135]. It has been shown that these filter-based approaches only achieve partial shift-invariance [16].

Local correlations are the reasons for the well-known advantages of extracting and combining local features before recognizing spatial or temporal objects. This is because configurations of neighboring variables can be classified into a small number of relevant categories (e.g. edges, corners). Convolutional Networks enforce the extraction of local features by restricting the receptive fields of hidden units to be local.

All these observations suggest that the integration of the CS method into CNNs has the potential to positively contribute to their performances. As already mentioned in section 3.2, the CS method protects local information in images, while emphasizing the edge information and decreasing the noise.

To experimentally test our approach, we have used three different classification architectures: AlexNet, DenseNet, and ResNet. We conducted experiments by applying the CS method to the channel-wise and element-wise approaches in order to analyze their effectiveness. In particular, [7] demonstrates that AlexNet is significantly less shift-invariant than the other architectures in practice. We have started the comparison of AlexNet's accuracy to that of the CS method applied at the beginning of this section. To understand the effect of the CS method on channel-wise concatenation we have employed the DenseNet algorithm and for the element-wise ones we have used the ResNet algorithm.

5.1 AlexNet

AlexNet is an image classification network where the input is an image of one of 1000 different classes such as cats, dogs, elephants, etc. It was the winner of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)-2012. The model consists of deep learning techniques such as strided convolution, pooling, augmentation, and dropout. In our study, we applied the CS method to AlexNet with the aim to analyze the effect of the CS method in the classification of a basic convolutional network.

5.1.1 Datasets

CIFAR-10 has been used as a starting point for practicing how to develop and evaluate convolutional deep learning neural networks for image classification. This dataset served as a well-established, widely used image classification dataset [136]. CIFAR-10 dataset consists of 32×32 sized 60,000 color images in 10 classes, with 6,000

images per class. These classes are completely mutually exclusive. There are 50,000 training images and 10,000 test images.

5.1.2 Architecture

AlexNet model is a deep learning model designed based on CNN architecture. The architecture of AlexNet consists of five convolutional layers, three max-pooling layers, two normalization layers, two fully connected layers, and one softmax layer in its architecture. AlexNet has some important features such as ReLU nonlinearity, overlapping pooling, data augmentation, and dropout [46]. ReLU is applied after every convolutional and fully connected layer. Dropout is applied before the first and the second fully connected layers. Main reason that we used this model is the fact that it includes different strided convolution layers, max, and average poolings. These layers are eligible for improvement with CS and allow us to evaluate the results of our experiments. We integrated the CS method into the first convolutional layer of this model.

5.1.3 Evaluation Metrics

The most often used metric for classification is accuracy. It is the ratio of the number of correct predictions to the total number of input samples. In addition to the classification accuracy, the loss function is also used for our test purpose. The difference between the prediction and the actual value is calculated as a loss value and it also indicates how well an algorithm is training.

5.1.4 Results

Firstly, we trained the AlexNet model for 1000 epochs and visualized it on Tensorboard. Accuracy, as well as loss, flattened out after 200 epochs. Because of this reason, it is suitable to run up to 200 epoch training for a 0.001 learning rate, 200

epochs for each run-through, 500 batches for each training epoch, 100 batches for each validating epoch, and 100 images for each training and validating batch.

We have integrated the CS method into the first layer of AlexNet using different padding techniques, different shifting steps, different aggregate functions, and different augmentation alternatives. We have observed that applying the CS method to the AlexNet algorithm gets better accuracy in all cases, as seen in Table 2, summarizing the results without augmentation, and Table 3, with augmentation.

Table 2: AlexNet Accuracy Results for CIFAR10 Dataset Without Augmentation.

Reflect padding | Constant padding | Replicate padding | Circular padding

AlexNet Accuracy Results for CIFAR10 Dataset Without Augmentation												
Original	**70.8**											
type	**MEAN**				**MAX**				**MEDIAN**			
shift												
1	72.7	72.4	71.7	72.0	**73.8**	71.0	72.2	72.6	72.1	72.0	72.3	71.5
2	71.6	73.1	71.7	72.4	71.4	70.4	71.8	71.6	71.9	71.0	71.8	71.3
3	71.7	72.0	72.5	72.2	71.5	70.7	71.8	72.0	71.2	71.3	71.1	72.4
4	71.0	71.8	73.7	72.7	71.5	72.2	72.6	71.7	71.8	71.8	72.4	71.3
1+2	71.8	71.1	71.7	71.7	72.7	71.3	71.1	71.5	73.3	72.2	71.1	71.3
1+2+3	71.1	71.6	72.7	71.8	71.0	70.9	71.9	72.1	71.0	71.7	71.8	72.1

Inside a neuron, the weights and the inputs are aggregated into a single value. For this purpose, fixed aggregation functions like sum or maximum are used to compute representations over data in deep learning algorithms. Also, in the original CS method studies mean is used as an aggregation function for combining different shift results [1]. There are many aggregation functions but we focused on a few important ones: Mean, median, and maximum. While mean and median measure the central tendency, the maximum shows extremum. The median and mean will be the same if the data sample is perfectly symmetrical and distributed. If the data is skewed, it may be more useful to calculate the median, which is less sensitive to outliers and extreme values. Results in Tables 2 and 3 show the results vary in relation to the amount of CS shift, padding method, and aggregation strategy. It can be seen that the mean aggregation

function results are more consistent and mostly higher than the others. For this reason, we used the mean aggregation function in our other experiments.

Table 3: AlexNet Accuracy Results for CIFAR10 Dataset With Augmentation.

Reflect padding | Constant padding | Replicate padding | Circular padding

AlexNet Accuracy Results for CIFAR10 Dataset With Augmentation												
Original	**74.7**											
type	**MEAN**				**MAX**				**MEDIAN**			
shift												
1	76.3	75.2	75.4	75.7	75.8	74.6	75.7	74.7	74.6	76.1	76.7	75.7
2	76.3	76.6	75.2	75.8	75.8	76.3	75.8	75.4	75.6	75.9	76.3	75.0
3	75.1	75.2	75.5	76.1	75.5	75.6	75.7	75.1	75.5	75.8	75.9	75.2
4	74.8	76.1	75.2	75.7	75.8	76.0	76.6	75.5	75.2	75.7	75.2	76.3
1+2	76.4	75.2	76.0	76.5	75.6	75.5	75.8	75.8	76.1	75.3	75.2	75.8
1+2+3	**77.0**	74.8	75.5	76.2	75.8	74.7	75.9	75.2	75.2	74.8	74.6	**77.0**

The other two factors that can affect the CS method's employment are padding and shifting, which are explained in section 3.2 in detail. The experimental results of reflect, constant, replicate, and circular padding types applied on the CS method are also shown in Tables 2 and 3. Although there are differences between the padding methods, it has been seen that the CS method is not highly sensitive to the padding type. We have applied the CS method with 1,2,3,4-pixel shifts and also recursive versions such as 1+2 and 1+2+3. The notation 1+2 means 1-pixel shift and 2-pixel shift applied image's convolution results are used. Similarly, 1+2+3 means 1, 2, and 3-pixel shifts applied to the convolution results of the images are used for the CS method.

All these experiments of classification algorithm trained on CIFAR-10 data set in AlexNet revealed that there are no big differences between the different padding methods, and we observed that small shift operations are more successful with the CS method, and this is up to 2 shifts maximum. This was expected, as in [1], it had been reported that the CS method results are highly dependent on the number of shifts, and 2-shift yields the best result. We have also seen that the integration of the CS method into the AlexNet algorithm with augmentation or with-

out augmentation generally produces higher accuracy. Code is publicly available at `https://github.com/UlkuUZUN/cs-alexnet-pytorch`.

5.2 DenseNet

The DenseNet (Dense Convolutional Network) is another important attended deep learning classification algorithm [137] used in 2017 CVPR and received the Best Paper Award. Similar to AlexNet, in the DenseNet model we have also chosen the CIFAR-10 dataset on the DenseNet Python codes [136]. Code is publicly available at `https://github.com/UlkuUZUN/cs-densenet-pytorch`.

5.2.1 Architecture

In DenseNet, concatenation is used, i.e., for each layer, the feature maps of all the preceding layers are used as inputs. As a result of this architecture, the vanishing-gradient problem is alleviated, feature propagation is empowered, feature reuse is increased and the number of parameters is reduced [138].

As seen in Figure 29, in DenseNet architecture, convolution and pooling operations are done by downsampling the exterior of the dense block, and feature concatenation (collective knowledge) is done in the interior of the dense block [138]. The transition blocks used in the DenseNet architecture consist of a batch-norm layer, 1×1 convolution followed by a 2×2 average pooling layer. We have integrated the CS method into the first convolution layer.

Figure 29: Block diagram of DenseNet classification algorithm.

5.2.2 Results

In this experiment, the original and different amounts of shift-applied versions of the DenseNet model have been trained up to 300 epochs. Similar to the AlexNet experiment, we have applied the CS method with 1,2,3,4-pixel shifts and also recursive versions such as 1+2 and 1+2+3 to the DenseNet model. The notation 1+2 means that the 1-pixel shift and 2-pixel shift applied image's convolution results are used. Similarly, 1+2+3 means 1, 2, and 3-pixel shifts have been used for the CS method. Additionally, augmented and without augmented versions of the model are evaluated separately. Again similar to the AlexNet experiment different padding alternatives (reflect, constant, replicate, circular) have been tried for the DenseNet algorithm. Testing has been done on CIFAR-10 to compare the accuracy of results.

Table 4: DenseNet Accuracy Results for CIFAR10 Dataset Without Augmentation.

Reflect padding | Constant padding | Replicate padding | Circular padding

DenseNet Accuracy Results for CIFAR10 Dataset Without Augmentation												
Original	**92.6**											
type	**MEAN**				**MAX**				**MEDIAN**			
shift												
1	93.1	93.2	92.9	92.8	92.9	92.9	93.0	92.9	**93.1**	92.9	92.9	92.9
2	93.1	93.2	**93.3**	92.9	92.9	93.0	92.6	92.9	92.8	93.0	93.0	93.0
3	93.0	93.1	92.9	93.1	92.9	92.9	**93.2**	92.8	93.0	92.9	**93.1**	**93.1**
4	92.6	93.0	92.8	93.0	93.0	93.1	92.9	92.9	92.9	92.8	92.9	92.9
1+2	93.1	93.0	92.9	93.1	93.0	93.0	93.0	92.8	**93.1**	92.9	92.8	92.7
1+2+3	92.9	92.5	92.9	92.8	92.9	93.0	92.7	92.7	93.0	92.9	93.0	92.9

Tables 4 and 5 show that integration of the CS method to DenseNet model, provides better results than the baseline model (though with a more limited amount compared to the AlexNet). This is a positive prospect for the CS method when we consider the different and more complex architecture of the DenseNet with its "collective knowledge" topology. We have shown that the CS method does not distort the algorithm with collective knowledge characteristics (dense blocks) and dropouts.

Table 5: DenseNet Accuracy Results for CIFAR10 Dataset With Augmentation.

Reflect padding | Constant padding | Replicate padding | Circular padding

DenseNet Accuracy Results for CIFAR10 Dataset With Augmentation												
Original	**95.3**											
type	**MEAN**				**MAX**				**MEDIAN**			
shift												
1	95.4	95.2	95.3	**95.5**	**95.5**	95.3	**95.5**	95.3	95.2	95.1	**95.4**	95.0
2	95.4	95.4	95.4	95.3	95.4	95.0	95.1	95.2	95.3	**95.4**	95.2	**95.4**
3	95.2	95.4	95.4	95.2	95.3	95.1	94.7	95.4	95.2	95.3	95.2	**95.4**
4	95.2	95.2	**95.5**	95.4	95.2	95.4	95.2	95.3	95.2	95.2	95.2	95.2
1+2	94.6	95.1	95.2	95.2	95.3	**95.5**	95.1	95.1	94.6	95.0	95.2	95.3
1+2+3	95.1	94.9	95.3	95.4	95.3	95.2	95.4	95.2	95.0	95.2	95.3	95.3

5.3 ResNet

Another popular classification algorithm is ResNet [139]. We have chosen this algorithm, on the grounds that adding layers together will not increase the network depth. Deep networks are difficult to train due to the well-known vanishing gradient problem. As the network grows larger, its performance becomes saturated, if not drastically degraded [50]. But, ResNet allows training of hundreds or even thousands of layers while still achieving excellent results [42].

5.3.1 Dataset

The ImageNet dataset [76] has more than 14 million images, hand-labeled across 20,000 categories. Also, unlike the CIFAR-10 dataset, the images in ImageNet are of decent resolution (224 x 224) and that's what poses a challenge for us: 14 million images, each 224 by 224 pixels. Processing a dataset of this size requires a great amount of computing power in terms of CPU, GPU, and RAM.

The CS method is a well-known denoising method. Because of this reason for the ResNet algorithm testing process, we have used ImageNet-C, which contains systematically corrupted ImageNet images. ImageNet-C contains impulse noise, defocus

and glass blur, simulated frost and fog, and various digital alterations of contrast, elastic transformation, pixelation, and jpeg compression [139]. Examples from different datasets of ImageNet-C. Our choice was the "contrast" dataset of ImageNet-C. This was chosen thinking that the application of CS to this dataset will give us a good understanding of its performance of corruption robustness.

5.3.2 Architecture

The ResNet architecture has got many variants such as ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-110, ResNet-152, ResNet-164 and ResNet-1202. Each of these has the same foundation but with a different number of layers. In this study, we have used ResNet-101 which performs the initial convolutions with 7×7 kernel sizes. Afterward, “conv2x” first includes one 3×3 pooling, and then contains 3 building blocks. Therefore, “conv2x” contains 3*3=9 convolutional layers. Similarly, “conv3x” contains 4 residual modules, which means that it contains 3*4=12 convolutional layers. “conv4x” contains 3*23=69 convolutional layers and “conv5x” contains 3*3=9 convolutional layers. The last part of ResNet-101 architecture contains a fully connected layer. We have integrated the CS method to the first convolution layer.

In [7], the authors aim to achieve an anti-aliasing effect by adding different filters before the downsampling process in deep-learning algorithms. They applied filters – Rect-2, Tri-3, Bin-5. In our experiments, it is observed that adding low-pass filtering to ResNet101 increases the performance by +0.6%. Because of this reason we also compared classification accuracy results with this work. Accuracy of ResNet-101 with Rect-2 filter named as low-pass filter-2 (lpf2), with Tri-3 filter named as low-pass filter-3 (lpf3) and with Bin-5 filter named as low-pass filter-5 named as lpf5.

5.3.3 Results

We evaluated the classifier performance using the Imagenet-C dataset. We have also run experiments using the low-pass filter solution in [26] to comparatively evaluate

the results. Because of this reason, we trained on the same Python codes [1]. All results are summarized in Table 6.

In our experiments for all model types in question, i.e. AlexNet, DenseNet, and ResNet, an increase has been observed in the results for mean, max, and median aggregation functions up to a 3.0 increase in accuracy.

The results of mean, max, and median aggregation are not conclusive and do not show that any of them stands out in general terms. However, we preferred to continue our experiments with the mean aggregation method, as it gives more consistent results in general.

Similarly, we evaluated CS results with/without augmentation settings and observed improvements in both. With the application of augmentation and the CS method together, an increase of up to 2.3 was observed in the results.

Table 6: Accuracy Results of ResNet101 trained for ImageNet-Contrast 50000.

	Original	Lpf2 filter	Lpf3 filter	Lpf5 filter	1-shifted CS
Resnet-101 accuracy	83.69	83.60	83.70	83.66	**83.78**

According to our experiments, the application of CS has improved the classification results for a variety of network models. Because of this reason we have also worked on integrating it into the object-detection algorithms.

[1] Source: GitHub - adobe/antialiased-cnns

CHAPTER 6

INTEGRATING CS INTO CNN-BASED MODELS FOR OBJECT DETECTION

In this chapter, we describe integration of the CS method into object detection algorithms and provide experimental results. Based on our literature survey, we have chosen YOLOv5, EfficientDet, and CenterNet algorithms to adapt the CS method. These algorithms have 3 main sections: Backbone, neck, and head. We have adapted the CS method on the first convolution layer and we presented results for widely used datasets such as Cheetah-Human, MS COCO2017, Dataset for Object Detection in Aerial images (DOTA), and Visdrone2019. PyTorch has been used in all our implementations as an open-source deep learning library [140].

6.1 YOLOv5

YOLO (You Only Look Once) family of models is widely used in object detection. In this dissertation, we experimented with YOLOv5 on many datasets with different structures. In each experiment, we adapted the CS method in the same way and analyzed the results by changing the datasets without changing the algorithm and parameters. The scope of the 4 datasets we are working on is explained in the "Datasets" section, and the details of the YOLOv5 algorithm are explained in more detail in the "Architecture" section. After the measurement techniques, we use for object detection are explained in the "Evaluation metrics" section, the results we obtained for each dataset are explained in the "Results" section. Code is publicly available at `https://github.com/UlkuUZUN/cs-yolov5`.

6.1.1 Datasets

6.1.1.1 Cheetah-Human thermal dataset

The cheetah-Human dataset is a small-scale dataset for cataloging animals with a trail camera, gathering statistics on wildlife behavior, or experimenting with other thermal and infrared imagery. The dataset was obtained from trail camera videos and it consists of 640×640 images.

We adopted the Cheetah-Human dataset's own splitting schema (90 train, 25 validation, and 14 test images) of 129 images with 231 annotations (186 cheetahs, 45 humans) [141]. This is because of the fact that subset data to train, validate, and test is extremely critical to ensure the prevention of overfitting, and to be able to have accurate evaluation results. The images in this dataset show continuity because they are obtained from video images and each image in the dataset is not densely packed with objects. Since it only focuses on two objects and the dataset size is small (only 90 images for training), it allows running experiments quickly. Because of this, we focused on this dataset to run our initial experiments.

6.1.1.2 COCO (COmmon Objects in Context) dataset

COCO is large-scale object detection, segmentation, and captioning dataset [123]. COCO-2017 object detection dataset with 80 object classes (116408 images for training and 5000 images for validation) consists of images with a size of 640×640.

We used the COCO-2017 dataset in our CS method experiments which will be explained in further sections because the COCO dataset is commonly used for benchmarking in publications and contains a considerable number of small objects in the dataset. Small objects make up 41.43% percent of all objects in the COCO-2017 dataset, and they occur in 51.82% of all images. Medium-sized objects are 34.32% of all objects, 70.07% of all images are medium-sized, again 24.24% of all objects, and 82.28% of all objects are large-sized. In addition to the small objects being dense, the

small objects occupying only 1.23% of the area also make the COCO-2017 dataset challenging ([142]).

The COCO-2017 dataset was used, again adhering to the splitting ratio of Roboflow [143], both because it is a widely studied dataset and to evaluate CS in challenging situations.

6.1.1.3 Dataset for Object Detection in Aerial Images (DOTA)

Because object detection in natural scenery only considers the scale, direction, and form of object samples on the earth's surface, to have a complementary analysis, aerial imaging datasets have also been used. Images in DOTA dataset were collected from the Google Earth, GF-2, and JL-1 satellites. These are provided by China Centre for Resources Satellite Data and Application and other aerial images are provided by CycloMedia B.V. DOTA [144].

DOTA-v1.0 contains 15 common categories, 2806 large-size images, and 188282 instances, including planes, ships, storage tanks, baseball diamonds, tennis courts, swimming pools, ground track fields, harbors, bridges, large vehicles, small vehicles, helicopters, roundabouts, soccer ball field and basketball court. Objects in different orientations occur in the images. In the dataset, each object is annotated by an oriented bounding box (OBB). In this annotation, vertices are arranged in clockwise order and apart from OBB, each instance is also labeled with a category and a difficulty index which indicates whether the instance is difficult to be detected or not (1 for difficult, 0 for not difficult).

In addition, there are a lot of small object instances. The aspect ratio and size of objects vary greatly, and extreme objects, such as bridges with extreme aspect ratios, can be discovered. The dataset is made more difficult by the huge disparities across items.

Since it is a challenging dataset that includes small, and large images with different orientations and aspect ratios, we also conducted experiments on the DOTA dataset.

Images in DOTA are so large (4000 x 4000 pixels) that they cannot be directly sent to CNN-based detectors [144]. Therefore, we crop a series of 1024 × 1024 pixel patches from the original images. So, some complete objects may be cut into two parts during the cropping process. In order to ensure that the training data and test data distributions approximately match, we randomly select 15749 images as the training set, 5297 images as the validation set, and 937 images as the testing set.

6.1.2 Architecture

YOLOv5 has four variants (YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5xl). Among these, we used the YOLOv5s, as there are no big differences between them except for the model layers/architecture and a number of parameters, and YOLOv5s can be trained relatively faster.

The YOLOv5 network consists of three main parts: Backbone, Neck, and Head. Backbone is a convolutional neural network that aggregates and forms image features. The neck is a set of layers that mix and combine image features to feed them to the prediction. The head takes the features from the neck and performs the box and class prediction steps.

6.1.3 Evaluation metrics

The performance metrics that have been used are Intersection over Union (IoU), Precision, Recall, F1 score, Average Precision (AP), and mean Average Precision (mAP). Details of these metrics are provided below.

6.1.3.1 Intersection Over Union (IOU)

Intersection over Union (IoU) is a ratio of the area of overlap between the predicted bounding box and the ground-truth bounding box over the area of union.

The predicted bounding boxes are compared with the ground truth bounding boxes by the detector according to IoU. The prediction is true when the IoU is larger than a set threshold. In other terms, true positive (correct detection) or false positive (wrong detection) are determined.

6.1.3.2 Precision and Recall

'True Positive (TP)', 'False Negative (FN)', and 'False Positive (FN)' detections are used in the calculation of precision and recall which in turn are used in the calculation of the F1 score. A confusion matrix is a table that is often used to describe the performance of a classifier on a set of test data for which the true values are known. Precision shows how many of the values we estimated as positive are actually positive. Precision, which is calculated as P = TP/ (TP +FP), is a good measure to determine especially when the costs of FPs are high.

Recall, which is calculated as R = TP/ (TP + FN), is a metric that shows how much of the operations we need to estimate as positive. A recall is a good measure to measure when there is a high cost associated with a False Negative.

6.1.3.3 F1-Score

F1-score is the harmonic mean of the precision and recall. The highest possible value of an F1-score is 1.0, indicating perfect precision and recall, and the lowest possible value is 0 if either the precision or the recall is zero. F1-Score is needed when a balance between Precision (P) and Recall (P) is sought for. F1 = 2*P*R / (P + R). The reason why it is a harmonic mean instead of a simple mean is that we should not ignore the extreme cases. If it was a simple average calculation, a model with a precision value of 1 and a recall value of 0 would have an F1 Score of 0.5, which would be misleading. The main reason for using the F1 Score value instead of accuracy is not to make an incorrect model selection in unevenly distributed data sets. In addition, the F1 Score is very important for us as we need a measurement metric that will include not only FP or FN but also all error costs.

Evaluating a model at different confidence levels also offers useful information about how the model is doing and what values improve model performance according to the design characteristics.

6.1.3.4 Mean Average Precision (mAP)

Average Precision is the mean of the precision scores after each relevant document is retrieved. The general definition for Average Precision is finding the area under the precision-recall curve. Precision and recall are always between 0 and 1. Therefore, AP falls within 0 and 1 accordingly. In addition to average precision and F1 score, the metric mean Average Precision (mAP) is extensively used in conventional object detection, which provides a performance evaluation in terms of regression and classification accuracies [145].

6.1.3.5 MS-COCO Method

Also, to calculate the results for the COCO dataset, a 101-point interpolated AP definition is used in the calculation. AP@ [.5:.95] corresponds to the average AP for IoU from 0.5 to 0.95 with a step size of 0.05. For the COCO competition, AP is the average over 10 IoU levels on 80 categories. Some other metrics collected for the COCO dataset are also calculated for evaluation. Additionally, using the pycocotools library (Official APIs for the MS-COCO Dataset, 2021) COCO individually evaluates AP at 0.5 and 0.75 IoU thresholds, this is denoted by AP@ 0.50 and AP@ 0.75 respectively. The performance is evaluated on AP in distinct object dimensions where AP (small) for small objects of area less than or equal to 32 pixels, AP (medium) for the medium object of area 32 < area < 96 pixels, AP (large) for large objects with an area greater than 96 pixels. COCO evaluates the algorithms on AR across scales with AR (small), AR (medium), and AR (large). Furthermore, COCO uses an additional metric that bases the AR evaluation on the number of detections per image number, specifically AR (max Dets=1) given 1 detection per image, AR (max Dets=10) for 10 detections per image, and AR (max Dets=100) for 100 detections per image.

6.1.4 Results

Firstly, YOLOv5 has been trained on each dataset without any modification as a baseline. Then, under the same conditions, YOLOv5s has been trained with the CS method integrated into the model's first convolution process. Then, the test set of each dataset has been used for evaluation and all results of experiments have been reported in the following subsections.

6.1.4.1 Yolov5 Cheetah-Human dataset results

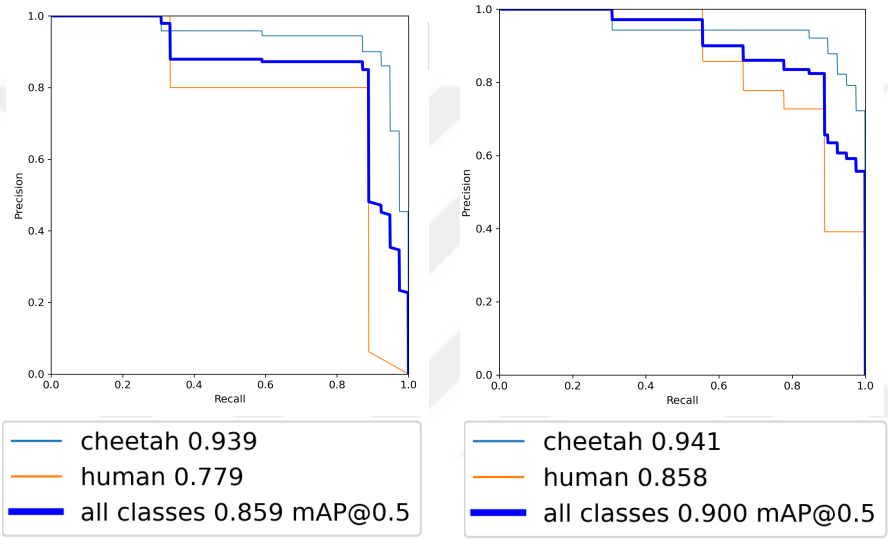
The object boundaries in the Cheetah-Human dataset are not very clear and as such, evaluation of this dataset allows investigation of the CS method on objects with relatively indistinct boundaries. An object detection model's performance at different levels of confidence may be usefully measured using precision and recall. The F1 score is also useful in identifying the level of confidence that best balances the precision and recall values for a specific model. As shown in Table 7, integration of the CS method into YOLOv5 provides better and more confident results.

Table 7: The detection performance of the CS integrated YOLOv5s on the Cheetah-Human dataset using different shift amounts.

	P	R	PR	F1 at Conf.
Base	94.2	94.0	85.4	82.0 at 33.1
CS_1	96.2	**99.0**	**93.2**	**87.0 at 46.9**
CS_2	**96.9**	99.0	**93.2**	88.0 at 43.3
CS_{1+2}	94.6	99.0	89.3	82.0 at 34.4

It can be seen in Table 7 that the CS method applied version of the model, YOLOv5s-CS, performs better than the original YOLOv5s model in terms of the validation results. Without any extra parameters, YOLOv5s-CS yields better mAP results for the model. Also, the precision (P) and recall (R) improve against the baseline. Especially, the 1-shift CS method increases PR value by 7.8 percentage points, against the base algorithm. It is noteworthy that the PR increase mainly due to an increase in recall.

Especially, according to the validation results, the cheetah mAP@.5 value increased from 0.939 to 0.941, while the mAP@.5 value for the human object increased from 0.779 to 0.858. In general, the average mAP@.5 results for all objects in the dataset increased from 0.859 to 0.900. It shows that the CS method makes the model more accurate in its detection. The CS method application increases both precision and recall values and it is seen that the difference between these two values also decreases. So, the CS method makes the model more confident. The precision-recall curve makes it easy to show the point where both the precision and recall are high.



(a) Original

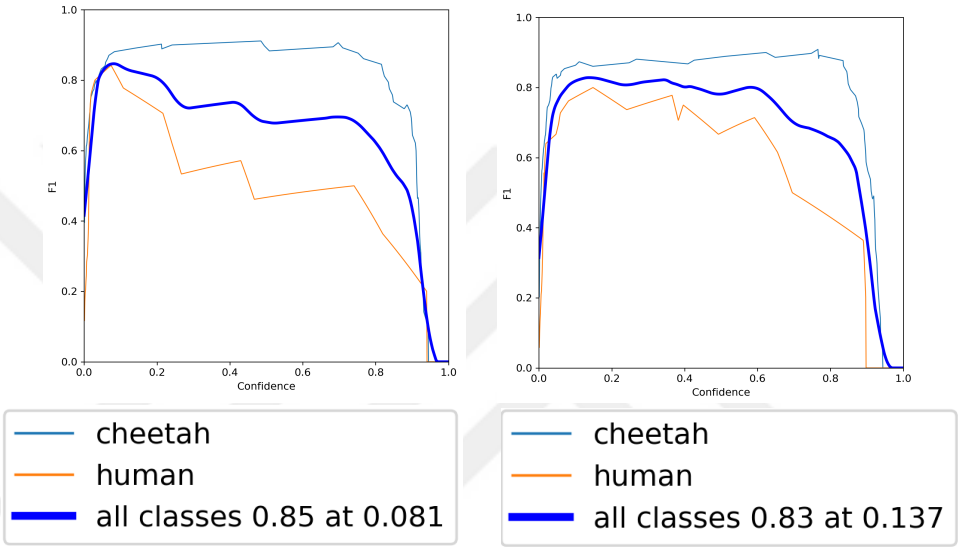
(b) 1 shift CS

Figure 30: Precision-Recall (PR) graphs of the Cheetah-Human validation dataset.

Figure 30 shows precision-recall graphs of the original YOLOv5s and YOLOv5s-CS. It is clear from the graphs of the CS method applied version precision-recall values increases in balance and with each step of the training this balance is preserved. To support our result, we also show Cheetah-Human dataset training F1 score graphs.

F1-confidence graph which is shown in Figure 31 is used in the YOLOv5 evaluation. In this graph, the vertical axis shows the F1-score and the horizontal axis shows the confidence value of the training results. The F1-confidence graph shown here, different from the previously recorded metrics, includes F1 scores for different confidence values. To show the stability of the results during training, different epoch stages are

plotted and shown in Figure 31. In Figure 31 it is seen that there have been increases in the CS method applied versions. For example, when we compare the training results of average scores, which are shown dark blue colored plot, the original model is far behind in each plot, with values of 0.85 at 0.081 while the YOLOv5 model with 1 shifted CS (the right- hand side graph) values have reached 0.83 at 0.137 confidence.



(a) Original

(b) 1 shift CS

Figure 31: F1 graphs of the Cheetah-Human validation dataset.

Samples of the Cheetah-Human dataset are shown in Figure 32. As seen in the upper part of Figure 32, the mAP value increases, or as in the example in the bottom part of Figure 32, it is seen that the objects (4 humans, 1 cheetah) labeled as in Figure 33 that could not be detected in the original model are detected after the CS adaptation. As seen in Figure 33, the format and detail of the dataset have been preserved in all experiments.

Because the Cheetah-Human dataset is so specific, small, and sparse dataset, we made experiments on some other datasets such as COCO, and DOTA.

Figure 32: Object detection results on the Cheetah-Human test set using baseline YOLOv5s (left) and YOLOv5s-CS_1 (right).

PIR-206_0_mov.txt

1	0.675	0.491	0.046	0.064
1	0.549	0.391	0.043	0.060
0	0.538	0.517	0.139	0.059
1	0.735	0.484	0.033	0.050
1	0.456	0.346	0.028	0.040

Figure 33: The Cheetah-Human test image class label detail. In the first column, 1 represents cheetah and zero is the human class label. The second and third columns are the xy coordinates of the bounding box. The fourth and fifth columns are the width and the height of the bounding box.

6.1.4.2 YOLOv5 COCO dataset results

We have trained the YOLOv5s model on the COCO2017 dataset with 929 targets on an NVIDIA Gigabyte GeForce GTX 1080. Our work investigates the CS method

application's impact on object detection algorithms. For this reason, before the subsampling, we have adapted the one and two-shift CS method just around the first convolution of the YOLOv5s model. After training, COCO val2017 dataset (1GB - 5000 images) has been evaluated with the pycocotools library. Table 8 shows the evaluation results of YOLOv5s on the COCO validation set in terms of Precision (P), Recall (R), Precision-Recall (PR), and F1-Confidence (Conf), metrics.

Table 8: Detection performance of Yolov5s-CS on COCO validation set using different shift amounts.

	P	R	PR	F1 at Conf.
Base	67.9	49.0	55.3	56.0 at 30.1
CS_1	67.1	49.0	**55.4**	56.0 at **32.3**
CS_2	66.5	49.0	54.1	56.0 at 30.1
CS_{1+2}	**69.2**	49.0	**55.7**	56.0 at **34.0**
CS_3	66.2	49.0	55.6	56.0 at **30.3**

It is assumed that there is a direct relationship between the confidence threshold and the precision. If the threshold is increased, the precision is also expected to increase while recall is expected to decrease. From the F1 curve, the confidence value that optimizes the precision and recall is represented as Conf. in this table. According to these results, CS_1 has better results than the base method, and CS_{1+2} has the overall best results. CS_3 does not offer any improvements over the baseline and comes with a higher computational cost so it is excluded in the remainder of the experiments. The difference between the baseline and CS method applied version is shown in Figure 34. In the results we obtained, an increase in the detection of objects is observed.

Table 9: Detection performance of YOLOv5s, YOLOv5s-CS_1, CS_2, CS_{1+2} on COCO validation set.

	YOLOv5s	YOLOv5s-CS_1	YOLOv5s-CS_2	YOLOv5s-CS_{1+2}
AP@0.50:0.95	37.2	**37.3**	37.2	**37.3**
AP@0.50	56.0	**56.1**	56.0	**56.3**
AP@0.75	40.5	40.5	40.5	**40.6**
AP@small	21.9	**22.0**	**22.0**	**22.0**
AP@medium	42.6	42.6	42.6	**42.7**
AP@large	47.3	47.3	47.2	**48.4**

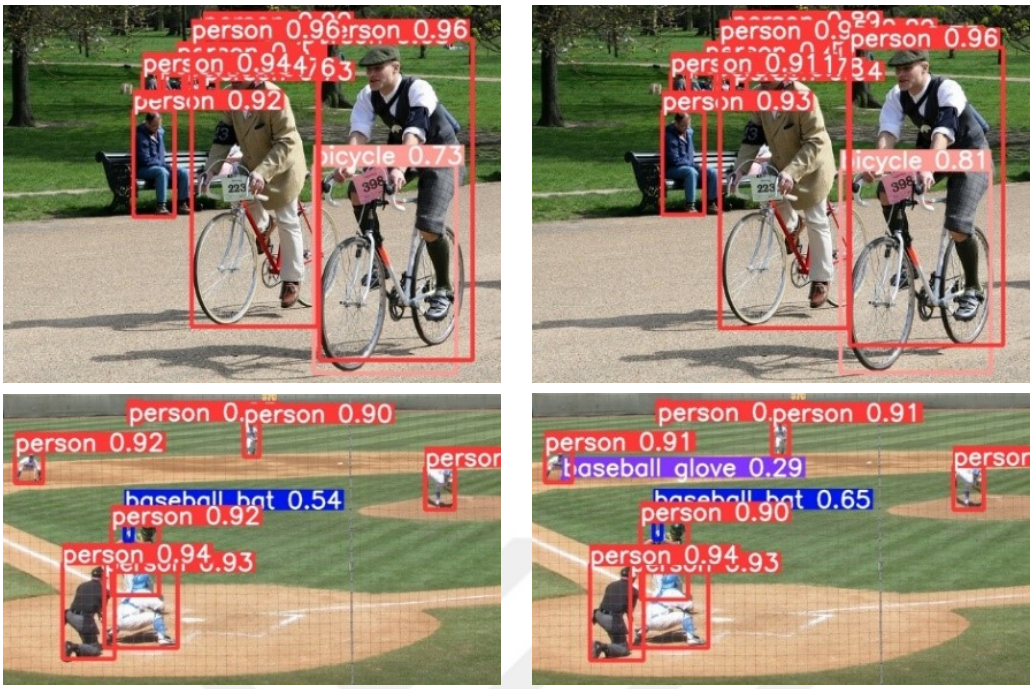


Figure 34: COCO test set results using base YOLOv5s model (left) and YOLOv5s-CS_1 (right).

As shown in Table 9, the proposed model improves over the baseline by 0.1, 0.3, and 0.1 percentage points on AP for small, medium, and large objects. The improvements in AR for small, medium and large objects are 0.1, 0.1, and 1.1, respectively. Due to the strong multi-scale ability, the model can cover a large range of receptive fields, boosting the performance on objects of different sizes. The highest increase in precision result of the CS method addition is seen on small objects. It is also important that this increase is obtained without decreasing the recall result of the small objects.

As seen from the representative examples in Figure 34 and the results in Table 8, Table 9, the CS method is effective for detecting small objects such as baseball gloves in the COCO dataset.

6.1.4.3 Yolov5 DOTA dataset results

In this section, we set groups of ablation experiments to evaluate the CS method application to the YOLOv5s model. We first show the impact of the CS method with

1-unit shift employment to the first convolution of the YOLOv5s model trained on the DOTA dataset and mAP results are presented. mAP scores for different object classes using baseline YOLOv5s and CS_1 variant, along with differential improvements in mAP are presented in Figure 35. The figure shows that the application of the CS method into YOLOv5 gives better results for detecting multiple categories. The results reveal that employing CS-convolution in YOLOv5 results in increasing performance throughout all object classes except small-vehicle, for which there is a negligible difference in performance.

In addition to mAP scores, the difference between the baseline and the CS method applied version is shown in Figure 35. This performance is important because DOTA dataset contains small, multi-scale objects in dense-targets scenes. We have evaluated the CS method on the DOTA dataset, to further analyze its performance in small object detection. The DOTA dataset has a wide range of image sizes, although the majority of the images are concentrated around the 1000–2500-pixel range.

For the measurement of object detection accuracy, we have compared the mAP results of the original and the CS-adapted versions of the model. CS integrated version of the model has an mAP (0.5) measure of 68.3 on the validation set, compared to 63.5 on the base model. In addition, compared with the original model, the overall precision of the original model increases by at least 4.8%.

We have trained the YOLOv5s model and CS variants on the DOTA dataset. Overall evaluation results are shown in Table 10 and Table 11.

Table 10: Detection performance of CS integrated YOLOv5s on DOTA validation set using different shift amounts.

	P	R	PR	F1 at Conf.
Base	93.1	85.0	63.5	65.0 at 39.1
1-shift CS	93.4	**88.0**	**68.3**	**68.0** at **47.7**
2-shift CS	93.2	82.0	60.8	62.0 at 45.3
1+2-shift CS	**94.0**	85.0	65.7	**66.0** at **49.7**

In Figure 36, the original YOLOv5 model and YOLOv5s-CS results on the DOTA dataset are shown. In order to provide visual evidence, Figure 36 reports the re-

Categories	Original YOLOv5s (mAP@0.5)	CS_1 applied YOLOv5s (mAP@0.5)	Difference
Small-vehicle	66.3	59.4	-6,9
Large-vehicle	83.7	83.9	0,2
Plane	89.0	91.8	2,8
Storage-tank	64.3	69.3	5,0
Ship	85.3	86.7	1,4
Harbor	78.1	82.6	4,5
Soccer-ball-field	44.5	62.0	17,5
Ground-track-field	41.8	50.6	8,8
Tennis-court	90.7	93.6	2,9
Swimming-pool	55.8	61.4	5,6
Baseball-diamond	71.1	74.7	3,6
roundabout	51.3	59.5	8,2
Basketball-court	54.2	60.8	6,6
Bridge	37.8	41.6	3,8
Helicopter	38.3	45.6	7,3
All classes	63.5	68.3	4,8

Figure 35: Comparison of the original YOLOv5s and YOLOv5s-CS_1 version (mAP@0.5) on the DOTA test dataset.

Table 11: Detection performance of YOLOv5s, YOLOv5s-CS_1, CS_2, CS_{1+2} on DOTA validation set.

	YOLOv5s	YOLOv5s-CS_1	YOLOv5s-CS_2	YOLOv5s-CS_{1+2}
AP@0.50:0.95	36.7	**41.4**	36.2	**39.5**
AP@0.50	63.5	**68.3**	60.8	**65.7**
AP@0.75	46.0	51.5	44.9	**49.2**

sults of object detection on the DOTA dataset after testing. It is possible to notice that the proposed approach permits identifying several objects in the images which suggests that producing higher-resolution feature maps simultaneously utilizing low-level features and high-level features is very critical to enable us to detect small remote sensing objects. At the same time, the CS method applied version of the model prevents disturbances that will cause false alarms on the detector, as shown in Figure

36. These scenes often contain objects with similar geometric construction, such as a long straight line.

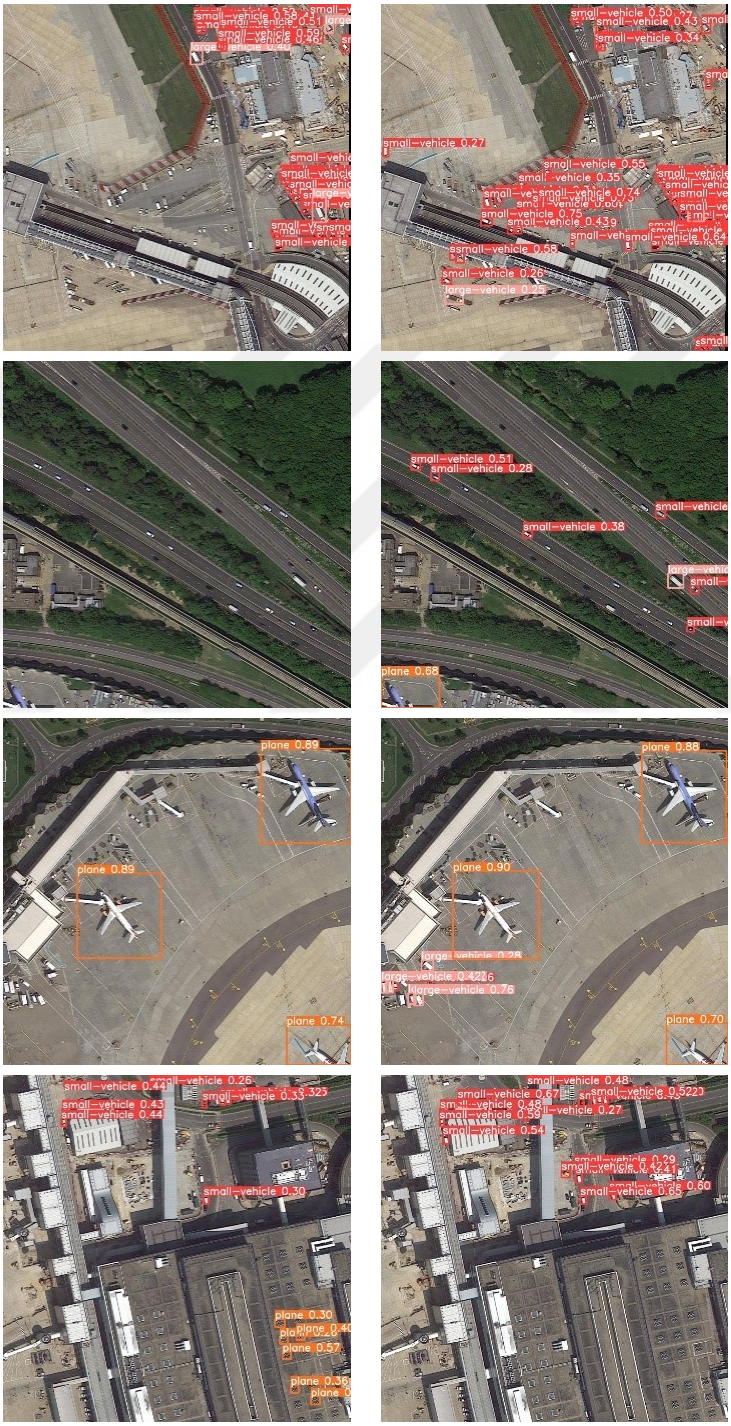
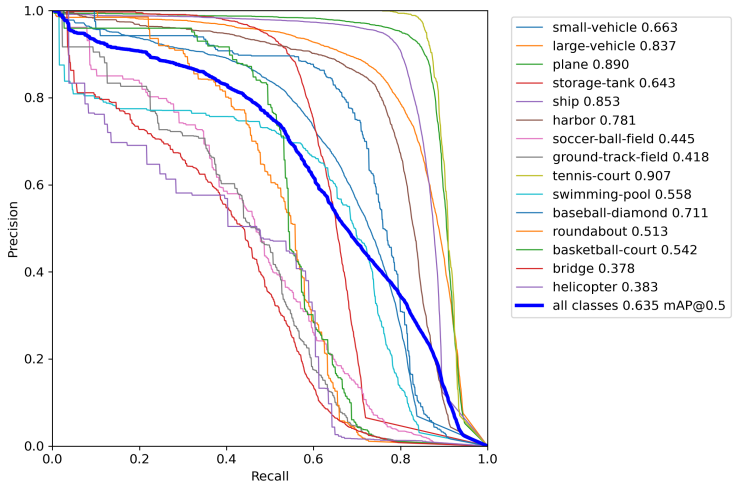
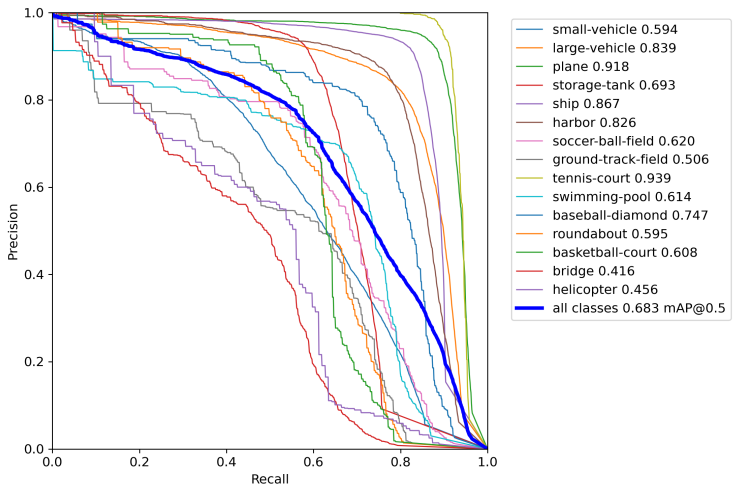


Figure 36: DOTA test set results using base YOLOv5s model and YOLOv5s-CS_1.

Figure 38 shows the F1 scores vs. confidence values for different classes as well as combined results for all classes (bold blue curve). The results indicate that CS integrated version has better performance and there is an improvement of 3% in terms of the F1 score of all classes and a 9% improvement in confidence values. F1-consistence graphs denote a much more stable result for the CS method applied version of YOLOv5. Overall F1-score results also increased from 0.416 to 0.482.



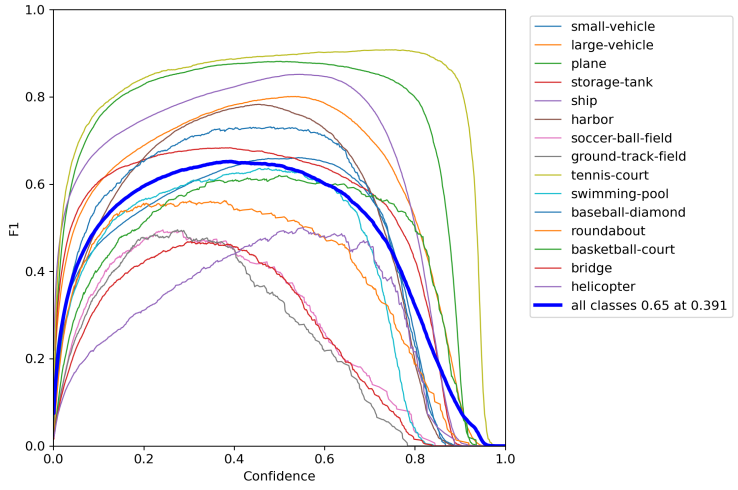
(a) Original



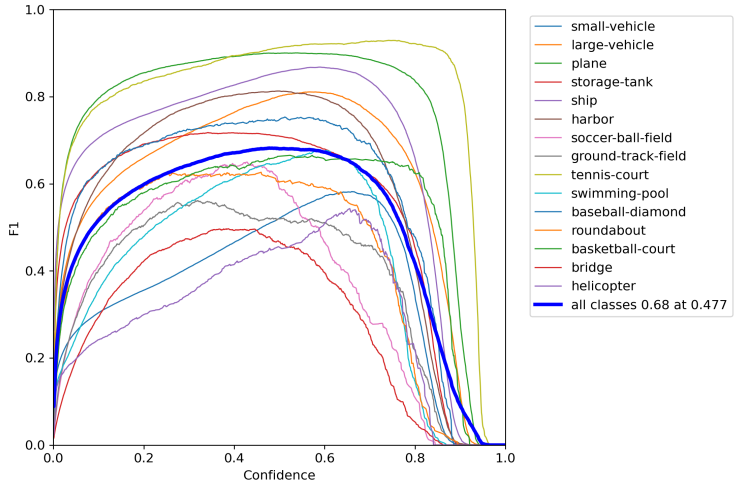
(b) 1 shift CS

Figure 37: Precission-Recall (PR) graph of the YOLOv5s (top) and YOLOv5s-CS_1 trained on the DOTA dataset.

Also when we look at the precision-recall graphs in Figure 37, we see that the values remain stable in general or that the overall average increases in small increments. Figure 37 shows accuracy (mAP) results for the models on the validation set. It is observed that the CS method applied version of the YOLOv5 model performed better than the original model. Applying the CS method helps to achieve much better validation accuracy with an up to 8% increase. The results show that a higher number of objects are detected by the CS-integrated version.



(a) Original



(b) 1 shift CS

Figure 38: F1 graph of the YOLOv5s (top) and YOLOv5s-CS_1 trained on the DOTA dataset.

6.2 EfficientDet

After working on the YOLOv5 object detection model, we also experimented with the EfficientDet object detection model [69] which builds on scaling neural networks (EfficientNet) and incorporates a novel bi-directional feature network (BiFPN) and new scaling rules. In this dissertation, we experimented with an improved model EfficientDet on many datasets. In each experiment, we adapted the CS method in the same way and analyzed the results by changing the datasets without changing the algorithm and parameters. The scope of the COCO dataset we are working on is explained in the "Dataset Selection" section, and the details of the EfficientDet algorithm are explained in more detail in the "Architecture" section. After the measurement techniques, we use for object detection are explained in the "Evaluation metrics" section, the results we obtained for each dataset are explained in the "Results" section.

6.2.1 Dataset selection

The COCO dataset is well-understood by state-of-the-art neural networks for object identification applications. Its adaptability and multi-purpose picture variety make it ideal for training and benchmarking computer vision models. Therefore, the COCO dataset, which we explained in detail in the YOLOv5 model, is also trained for the EfficienDet model.

6.2.2 Architecture

The EfficientDet is made up of three main parts: A framework for extracting features from an image. A feature network that takes multiple levels of features as input from the backbone and outputs a combined list of features representing the basic features of the image. And the last is a class/box network that uses the combined function to predict the class and location of each object [69].

Most of the previous detectors simply used top-down pyramid networks (FPNs), but top-down FPNs are inherently limited to one-way traffic. Alternative FPNs such as PANet add extra upstream at an extra computational cost. Recent attempts to use Neural Architecture Search (NAS) have uncovered a more sophisticated NASFPN architecture.

6.2.3 Results

6.2.3.1 EfficientDet COCO Dataset results

We evaluate the performance of CS method adaptation to object detection algorithms EfficientDet on COCO2017 datasets. We have integrated the CS method before subsampling at the first convolution of the EfficientDet model. After 300 epoch training, the COCO val2017 dataset has been evaluated with pycocotools library. Evaluation results are shown in Table 12.

Table 12: Detection performance of EfficientDet, EfficientDet-CS_1, CS_2, CS_{1+2} on COCO validation set.

	EfficientDet	EfficientDet-CS_1	EfficientDet-CS_2	EfficientDet-CS_{1+2}
AP@0.50:0.95	26.4	27.9	**28.1**	**28.1**
AP@0.50	43.6	44.8	**44.9**	**44.9**
AP@0.75	27.7	29.3	**29.7**	29.6
AP@small	3.1	**3.5**	**3.5**	**3.5**
AP@medium	25.5	**27.3**	27.0	27.1
AP@large	43.2	45.1	**45.9**	**45.9**

The model performance improves over its original version by 0.4, 1.8, and 2.7 percentage points on AP for small, medium, and large objects. Due to the strong multi-scale ability, the model can cover a large range of receptive fields, boosting the performance on objects of different sizes. The highest increase in precision result of the CS method addition is seen on small objects. It is also important that this increase is obtained without decreasing the recall result of the small objects.

Sample visual results on the test set can be seen in Figure 39. When the two images on the top are compared, it is seen that the base model detects the bus with 95% confidence, while CS integrated model has 97% confidence. Likewise, while it was 40% in the truck base model, it increased to 47% with the application of CS, and it is also seen that the person on the motorcycle, which was not detected using the base model, was detected with 47% confidence by the CS integrated model. When the images at the bottom are compared, although the confidence values of some objects have decreased, it is a remarkable improvement that small objects such as books are detected by the CS integrated version. Code is publicly available at `https://github.com/UlkuUZUN/cs-efficientdet`.

Figure 39: COCO test dataset results using EfficientDet base model and EfficientDet-CS_1.

6.3 CenterNet

6.3.1 Dataset selection

VisDrone2019 dataset has been used to analyze the performance of CS method on drone images. The benchmark dataset consists of 288 video clips made up of 261908 frames and 10209 static images that were taken by various drone-mounted cameras. These clips cover a variety of topics such as location (they were taken from 14 different cities in China that are thousands of kilometers apart), environment (urban and rural), objects (pedestrians, vehicles, bicycles, etc.), density, and environment (urban and rural) (sparse and crowded scenes)[146].

6.3.2 Architecture

CenterNet is an anchorless object detection model and does not require application of NMS (Non Maximum Suppression) at the post-processing stage. The majority of detectors encode their predictions using anchors and the generated feature map predicts many boxes for each spatial cell. Each box prediction is represented by x, y, width, and height offsets with respect to the appropriate anchor and the center of the cell, respectively. The idea that box predictions may be ordered for relevance based on where their centers are rather than where they overlap the object is the foundation for this method.

The CenterNet has two basic prediction heads: for confidence heat map, and for regression values. The confidence heat map is used to remove irrelevant predictions. The other head works to estimate the regression values for the box sizes and offsets that refer to the box center estimated using the heat map. So the heat map plays an important role in detecting an object.

The CenterNet provides different backbone models such as ResNet-18, DLA-34, and Hourglass-104. We have used the ResNet-18 backbone in our experiments. Because the ResNet-18 enables us to achieve fast results in accordance with our technical

infrastructure. In CenterNet model three transposed convolutional networks are added to the standard ResNet modules.

6.3.3 Results

6.3.3.1 CenterNet Visdrone Dataset results

Lastly, we have integrated the CS method into CenterNet with ResNet-18 backbone and trained on VisDrone2019-DET to evaluate the effect of CS on anchor-free algorithms.

The visDrone2019-Det dataset contains crowded and sparse small objects and the occlusion rate is higher than the other datasets we experimented with. In addition to mAP scores, the difference between the baseline and the CS method applied version is shown in Figure 40.

Categories	CenterNet (mAP@0.5)	CenterNet (mAP@0.5)	Difference
Pedestrian	**25.2**	25.1	-0.1
People	**16.9**	16.5	-0.4
Bicycle	7.1	**8.1**	1.0
Car	**54.1**	54.0	-0.1
Van	30.5	**31.7**	1.2
Truck	**18.0**	17.5	-0.5
Tricycle	13.7	**14.1**	0.4
Awning-tricycle	6.6	**7.3**	0.7
Bus	28.2	**32.3**	4.1
Motor	23.3	**23.5**	0.2
All classes	48.5	**48.8**	0.3

Figure 40: Comparison of the original and CS_{1+2} YOLOv5s in terms of mAP@0.5 values of objects in the VisDrone2019-DET validation dataset.

Table 13: Detection performance of CenterNet, CenterNet-CS_1, CS_2,CS_{1+2} on VISDRONE2019-DET validation set.

	CenterNet	CenterNet-CS_1	CenterNet-CS_2	CenterNet-CS_{1+2}
AP@0.50:0.95	24.2	**24.3**	24.2	**24.5**
AP@0.50	48.5	**48.6**	48.2	**48.8**
AP@0.75	20.7	21.1	21.0	**21.1**

In these results, it is seen that the detection of objects of different sizes such as buses, vans, and bicycles is positively affected by the CS convolution. On the other hand, it is slightly negatively affected by occlusion.

6.4 Discussion

Evaluations of different datasets show that the selection of the CS shift parameter is dependent on the dataset and the size of the objects in the dataset. If the objects are small, CS_1 is expected to have better performance, while for datasets having larger objects, CS_2 or CS_{1+2} are preferable. Specifically, the best results for DOTA which contains smaller objects work the best with CS_1 while the best results for COCO are obtained with CS_{1+2}.

CS requires the calculation of convolution for various shifted inputs and, as such, it increases the computational complexity. On the other hand, this increase is limited as we integrate CS into the first convolutional layer only and the remainder of the network is unchanged. In order to compare the computational effect of CS method adaptation in the models, the GFLOPs values were calculated using a FLOPS counter tool [147].

As seen in Table 14, while the number of operations in the first convolution layer of YOLOv5s, EfficientDet-D0, and CenterNet are approximately 0.46, 0.11, and 1.24 GFLOPS respectively, these numbers increase by a factor of 9 with CS_1 (8 more convolutions corresponding to different shifts in addition to the standard convolution) and this results in overall computational complexity of 14.28, 5.58 and 112.98 GFLOPS

Table 14: Computational complexity results at 512x512 input resolution.

Model		Params (M)	First Conv2D (GFLOPs)	Overall (GFLOPs)
Yolov5s	Base	7.2	0.46	10.58
	CS_1		4.16	14.28
	CS_{1+2}		7.86	17.98
EfficientDet-D0	Base	3.9	0.11	4.66
	CS_1		1.02	5.58
	CS_{1+2}		1.94	6.48
CenterNet-ResNet-18	Base	15.82	1.24	103.10
	CS_1		11.16	112.98
	CS_{1+2}		21.08	122.90

respectively, corresponding to 35%, 20%, and 10% higher computational complexity compared to their respective baselines. For CS_{1+2}, the computational complexity of the first convolutional layer increases by a factor of 17 and this results in overall computational complexity of 14.28, 5.58, and 112.98 GFLOPS respectively, corresponding to 70%, 39%, and 19% higher computational complexity compared to their respective baselines. According to the results, integration of the CS method to different network model types has different effects on the computational complexity and while applying multiple shifts imposes a certain computational complexity on the system, small shift steps can be tolerated, especially in CenterNet and EfficentDet networks.

In our experiments, although it has been seen that the employment of the CS method has been more successful in detecting small objects, we also noted that factors such as object size distribution (i.e. availability of both large and small objects in the scene) and kernel size have been affecting the success rates. In general, it has been observed that the success of the CS method decreased for shift amounts higher than 2 pixels, and the 1-shift CS method remained more stable in the algorithms and datasets tested. One of the frequent issues of deep learning algorithms is that sometimes the same model gives varied results [148]. The response surface across which the minimization occurs is nonconvex, and many CNN estimate approaches are sensitive to the optimizer's starting point, among other factors. Because of this reason, although

experiments are completed in the same environment and with the same parameters, it is observed that the results are unstable. One of the solutions to this problem probably is using mean and variance values of the experiment results. However, this is prohibitively expensive due to high computational requirements we reported the most stable result values in our dissertation and as such obtained consistent results as applied to different datasets and networks.

CHAPTER 7

CONCLUSION AND FUTURE WORK

In this work, we have investigated the adaptation of the cycle-spinning (CS) method into different deep-learning algorithms.

First, we have adapted the cycle-spinning (CS) method [1] to Generative Adversarial Network (GAN) for removal of raindrops from images for enhancing most of the adverse weather images and increase the quality of scenes in real-time [27]. The CS method has been shown to increase perceptual quality in terms of full-reference metrics. It also increases object detection performance, justifying its usefulness when used as a pre-processing operation under adverse weather conditions in intelligent transportation systems and outdoor surveillance systems. In addition, higher mAP scores indicate the potential of using the CS method for better object detection in other application areas [27]. The proposed method is generic in the sense that it can be integrated into a variety of image enhancement algorithms that use CNNs.

Second, the integration of the CS method into different CNN classification algorithms, namely, AlexNet, DenseNet, ResNet has been investigated. The experimental results have shown that the employment of the CS method results in increased consistency and accuracy.

Lastly, we have adapted the CS method to CNN-based object detection algorithms, namely, YOLOv5, EfficientDet, CenterNet. The CS method has been shown to increase object detection performance by emphasizing edges and smoothing noises, justifying its usefulness with easy adaptation into algorithms and without the need

for extra learnable parameters. The higher mAP scores clearly indicate the potential of using the CS method for better object detection in many other applications.

We have shown that the CS method is a useful tool especially when it is employed before downsampling layers in CNN-based algorithms. This is because it reduces signal degradation during sampling. As the proposed method is generic, it can be easily adapted to other image processing and computer vision problems.

Additionally, we investigated whether the CS method could be employed for the layers rather than for the whole algorithm as we did in the GAN study. We have experimented with raindrop degradation in GAN but this dataset also is trained for object detection. Furthermore, artifact types can be diversified with other artifacts such as rain streaks, fog, and random noises.

With these experiments, we have shown that the CS method can be adapted to different layers in different algorithm architectures and proved the flexibility of this method for adapting to different components of deep learning. Our tests also revealed the effects of different padding approaches used in the CS method.

This dissertation has opened the door to many new research topics. Investigation of whether the CS method can be added to the other layers or not, and whether different shift methods are effective or not on different datasets are prospective for future work. This study has aimed to adapt the CS method while keeping the general architectures of the models and their total number of learnable parameters. Because of this reason, it is not trained for fine-tuning and transfer learning.

In our studies, we have observed that CS method-adapted algorithms converge faster. Although we have shown that the employment of the CS method increases computational complexity, the fact that they are faster to converge has the potential to alleviate this limitation. In addition, smaller CS-enabled models have the potential to provide on-par performance against larger models, hence it may allow using smaller models, which is especially desired in deployment in low-power devices.

In addition, the issue of small input shifts is closely related to adversarial attacks. CNNs are vulnerable to such adversarial attacks [149]. Measuring the robustness of CS-enabled models against adversarial attacks is also a prospective topic.

REFERENCES

[1] A. Temizel and T. Vlachos, "Wavelet domain image resolution enhancement using cycle spinning and edge modelling," *13th European Signal Processing Conference, EUSIPCO 2005*, vol. 41, no. 3, pp. 177–180, 2005.

[2] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *ArXiv*, vol. abs/1905.05055, 2019.

[3] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization," *CoRR*, vol. abs/1610.02391, 2016.

[4] R. Qian, R. T. Tan, W. Yang, J. Su, and J. Liu, "Attentive Generative Adversarial Network for Raindrop Removal from A Single Image," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2482–2491, 2018.

[5] T. S. Nazaré, G. B. P. da Costa, W. A. Contato, and M. Ponti, "Deep convolutional neural networks and noisy images," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications* (M. Mendoza and S. Velastín, eds.), (Cham), pp. 416–424, Springer International Publishing, 2018.

[6] J. Lee, J. Yang, and Z. Wang, "What does cnn shift invariance look like? a visualization study," in *Computer Vision – ECCV 2020 Workshops* (A. Bartoli and A. Fusiello, eds.), (Cham), pp. 196–210, Springer International Publishing, 2020.

[7] R. Zhang, "Making Convolutional Networks Shift-Invariant Again," in *International conference on machine learning*, pp. 7324–7334, 2019.

[8] A. Azulay and Y. Weiss, "Why do deep convolutional networks generalize so poorly to small image transformations?," *Journal of Machine Learning Research*, vol. 20, pp. 1–25, 2019.

[9] S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," in *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, pp. 1–6, 2016.

[10] L. Engstrom, B. Tran, D. Tsipras, L. Schmidt, and A. Madry, "Exploring the landscape of spatial robustness," *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 3218–3238, 2019.

[11] Y. Zheng, Y. Chen, and M. Sarem, "Group-Teaching: Learning Robust CNNs from Extremely Noisy Labels," *IEEE Access*, vol. 8, pp. 34868–34879, 2020.

[12] C. Kanbak, S. M. Moosavi-Dezfooli, and P. Frossard, "Geometric Robustness of Deep Networks: Analysis and Improvement," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4441–4449, 2018.

[13] S. Gulshad and A. W. M. Smeulders, "Natural perturbed training for general robustness of neural network classifiers," *ArXiv*, vol. abs/2103.11372, 2021.

[14] A. Kumar and E. Amid, "Constrained instance and class reweighting for robust learning under label noise," *CoRR*, vol. abs/2111.05428, 2021.

[15] N. Li, Y. Chen, Z. Ding, and D. Zhao, "Shift-invariant convolutional network search," in *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, 2020.

[16] A. Chaman and I. Dokmanić, "Truly shift-invariant convolutional neural networks," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3772–3782, 2021.

[17] C. Xiao, J.-Y. Zhu, B. Li, W. He, M. Liu, and D. Song, "Spatially transformed adversarial examples," in *International Conference on Learning Representations*, 2018.

[18] A. Aydin, D. Sen, B. T. Karli, O. Hanoglu, and A. Temizel, "Imperceptible adversarial examples by spatial chroma-shift," in *Proceedings of the 1st International Workshop on Adversarial Learning for Multimedia*, ADVM '21, (New York, NY, USA), p. 8–14, Association for Computing Machinery, 2021.

[19] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 764–773, 2017.

[20] Z. Lin, Z. He, S. Xie, X. Wang, J. Tan, J. Lu, and B. Tan, "Aanet: Adaptive attention network for covid-19 detection from chest x-ray images," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 11, pp. 4781–4792, 2021.

[21] J. Chen, X. Wang, Z. Guo, X. Zhang, and J. Sun, "Dynamic region-aware convolution," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Los Alamitos, CA, USA), pp. 8060–8069, IEEE Computer Society, jun 2021.

[22] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.

[23] C. Vasconcelos, H. Larochelle, V. Dumoulin, R. Romijnders, N. L. Roux, and R. Goroshin, "Impact of Aliasing on Generalization in Deep Convolutional Networks," *IEEE/CVF International Conference on Computer Vision*, pp. 10529–10538, 8 2021.

[24] R. C. Gonzalez and R. E. Woods, *4TH EDITION Digital image processing*. 2018.

[25] R. R. Coifman and D. L. Donoho, *Translation-Invariant De-Noising*, pp. 125–150. New York, NY: Springer New York, 1995.

[26] H. Zhang, V. Sindagi, and V. M. Patel, "Image De-raining Using a Conditional Generative Adversarial Network," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–13, 1 2019.

[27] U. Uzun and A. Temizel, "Cycle-Spinning GAN for Raindrop Removal from Images," in *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, no. September, pp. 1–6, IEEE, 9 2019.

[28] U. Uzun and A. Temizel, "Cycle-spinning convolution for object detection," *IEEE Access*, vol. 10, pp. 76340–76350, 2022.

[29] M. A. Rodriguez-Hernandez, "Shift selection influence in partial cycle spinning denoising of biomedical signals," *Biomedical Signal Processing and Control*, vol. 26, pp. 64–68, 2016.

[30] I. V. Florinsky, "Chapter 5 - errors and accuracy," in *Digital Terrain Analysis in Soil Science and Geology (Second Edition)* (I. V. Florinsky, ed.), pp. 149–187, Academic Press, second edition ed., 2016.

[31] A. Nosratinia, "Denoising JPEG images by re-application of JPEG," *1998 IEEE 2nd Workshop on Multimedia Signal Processing*, vol. 1998-Decem, no. Figure 1, pp. 611–615, 1998.

[32] Coifman R.R. and Donoho, "Translation-Invariant Denoising," in *Lecture Notes in Statistics*, pp. 125–150, 1995.

[33] A. K. Fletcher, V. K. Goyal, and K. Ramchandran, "Wavelet Denoising by Recursive Cycle Spinning," *ICIP(2)*, pp. 873–876, 2002.

[34] P. Chen and D. Suter, "Shift-invariant wavelet denoising using interscale dependency," in *Proceedings - International Conference on Image Processing, ICIP*, vol. 5, pp. 1005–1008, 2004.

[35] A. Nosratinia, "Postprocessing of JPEG-2000 images to remove compression artifacts," *IEEE Signal Processing Letters*, vol. 10, no. 10, pp. 296–299, 2003.

[36] A. Nosratinia, "Enhancement of jpeg-compressed images by re-application of jpeg," *Journal of Vlsi Signal Processing Systems for Signal Image and Video Technology - J VLSI SIGNAL PROCESS SYST S*, vol. 27, pp. 69–79, 02 2001.

[37] A. Temizel and T. Vlachos, "Wavelet domain image resolution enhancement using cycle-spinning," *IET Electronics Letters*, vol. 41, no. 3, pp. 119–121, 2005.

[38] A. Temizel and T. Vlachos, "Wavelet domain image resolution enhancement," *IEE Proceedings: Vision, Image and Signal Processing*, vol. 153, no. 1, pp. 25–30, 2006.

[39] A. Temizel and T. Vlachos, "Image resolution upscaling in the wavelet domain using directional cycle spinning," *Journal of Electronic Imaging*, vol. 14, no. 4, 2006.

[40] B. N. Aravind and K. V. Suresh, "Multispinning for image denoising," *Journal of Intelligent Systems*, vol. 21, no. 3, pp. 271–291, 2012.

[41] E. Grossi and M. Buscema, "Introduction to artificial neural networks," *European Journal of Gastroenterology and Hepatology*, vol. 19, no. 12, pp. 1046–1054, 2007.

[42] K. He, "Deep Residual Learning for Image Recognition," 2015.

[43] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, *Review of deep learning: concepts, CNN architectures, challenges, applications, future directions*, vol. 8. Springer International Publishing, 2021.

[44] M. J. M. Mazack, "Algorithms for Handwritten Digit Recognition," *International Conference on Artificial Neural Networks (Paris).*, no. July, pp. 53–60, 1995.

[45] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 5987–5995, 2017.

[46] B. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2012.

[47] M. Amer and T. Maul, "A review of modularization techniques in artificial neural networks," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 527–561, 2019.

[48] M. Jaderberg, K. Simonyan, A. Zisserman, and k. kavukcuoglu, "Spatial transformer networks," in *Advances in Neural Information Processing Systems* (C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, eds.), vol. 28, Curran Associates, Inc., 2015.

[49] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5455–5516, 2020.

[50] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–14, 2015.

[51] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *32nd International Conference on Machine Learning, ICML 2015*, vol. 1, pp. 448–456, 2015.

[52] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," pp. 1–18, 2012.

[53] S. Gulshad and A. Smeulders, "Natural Perturbed Training for General Robustness of Neural Network Classifiers," 2021.

[54] A. Azulay, "Why do deep convolutional networks generalize so poorly to small image transformations ? arXiv : 1805 . 12177v2 [cs . CV] 18 Feb 2019," 2017.

[55] L. Engstrom, B. Tran, D. Tsipras, L. Schmidt, and A. Madry, "A Rotation and a Translation Suffice: Fooling CNNs with Simple Transformations," pp. 1–20, 2017.

[56] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, no. 1, pp. 586–595, 2018.

[57] C. E. Shannon, "Communication in the Presence of Noise," *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, 1949.

[58] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 580–587, 2014.

[59] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems* (C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, eds.), vol. 28, Curran Associates, Inc., 2015.

[60] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 386–397, 2020.

[61] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9905 LNCS, pp. 21–37, 2016.

[62] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 779–788, 2016.

[63] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 6517–6525, 2017.

[64] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," 2018.

[65] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv*, 2020.

[66] G. Jocher, A. Stoken, J. Borovec, NanoCode012, ChristopherSTAN, L. Changyu, Laughing, tkianai, A. Hogan, lorenzomammana, yxNONG, AlexWang1900, L. Diaconu, Marc, wanghaoyang0106, ml5ah, Doug, F. Ingham, Frederik, Guilhen, Hatovix, J. Poznanski, J. Fang, L. Yu, changyu98, M. Wang, N. Gupta, O. Akhtar, PetrDvoracek, and P. Rai, "ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements," 10 2020.

[67] E. Ur Rahman, Y. Zhang, S. Ahmad, H. I. Ahmad, and S. Jobaer, "Autonomous vision-based primary distribution systems porcelain insulators inspection using UAVs," *Sensors (Switzerland)*, vol. 21, no. 3, pp. 1–24, 2021.

[68] M. Mehdi Özel, "Drone Dataset (UAV)." https://www.kaggle.com/datasets/dasmehdixtr/drone-dataset-uav?select=drone_dataset_yolo. [Online; accessed 2021-12-18].

[69] M. Tan, R. Pang, and Q. V. Le, "EfficientDet," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 10778–10787, 2020.

[70] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 10691–10700, 2019.

[71] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 734–750, 2018.

[72] X. Zhou, J. Zhuo, and P. Krähenbühl, "Bottom-up object detection by grouping extreme and center points," in *CVPR*, 2019.

[73] M. Manfredi and Y. Wang, "Shift Equivariance in Object Detection," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12540 LNCS, pp. 32–45, 2020.

[74] I. J. Goodfellow, J. Pouget-abadie, M. Mirza, B. Xu, and D. Warde-farley, "Generative Adversarial Nets," pp. 1–9, 2014.

[75] J. Cheng, Y. Yang, X. Tang, N. Xiong, Y. Zhang, and F. Lei, "Generative Adversarial Networks : A Literature Review," vol. 14, no. 12, pp. 4625–4647, 2020.

[76] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84–90, 5 2017.

[77] T. Salimans, I. Goodfellow, V. Cheung, A. Radford, and X. Chen, "Improved Techniques for Training GANs," pp. 1–10, 2016.

[78] Y. J. Cao, L. L. Jia, Y. X. Chen, N. Lin, C. Yang, B. Zhang, Z. Liu, X. X. Li, and H. H. Dai, "Recent Advances of Generative Adversarial Networks in Computer Vision," *IEEE Access*, vol. 7, no. c, pp. 14985–15006, 2019.

[79] L. Jin, F. Tan, and S. Jiang, "Generative Adversarial Network Technologies and Applications in Computer Vision," *Computational Intelligence and Neuroscience*, vol. 2020, no. 1, 2020.

[80] R. Wu, G. Zhang, S. Lu, and T. Chen, "Cascade EF-GAN: Progressive Facial Expression Editing with Local Focuses," *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5021–5030, 2020.

[81] A. D.Donahue, "Adversarial Text Generation Without Reinforcement Learning," 2019.

[82] Y. Zhu, Y. Zhang, H. Yang, and F. Wang, "Gancoder: An automatic natural language-to-programming language translation approach based on GAN," *CoRR*, vol. abs/1912.00609, 2019.

[83] E. Brophy, Z. Wang, and T. E. Ward, "Quick and Easy Time Series Generation with Established Image-based GANs," pp. 1–8, 2019.

[84] D. Li, D. Chen, L. Shi, B. Jin, J. Goh, and S.-k. Ng, "MAD-GAN : Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks," pp. 1–17, 2019.

[85] J. Yoon and D. Jarrett, "Time-series Generative Adversarial Networks," no. NeurIPS, pp. 1–11, 2019.

[86] Q. H. Le, K. Youcef-toumi, D. Tsetserukou, and A. Jahanian, "GAN Mask R-CNN: Instance semantic segmentation benefits from generative adversarial adversarial networks," 2020.

[87] P. Luc, C. Couprie, J. Verbeek, and L. J. Kuntzmann, "Semantic Segmentation using Adversarial Networks," 2016.

[88] Z. Zhao, Y. Wang, and K. Liu, "Semantic Segmentation by Improved Generative Adversarial Networks," pp. 1–18, 2021.

[89] Q. Yan and W. Wang, "DCGANs for image super-resolution , denoising and debluring," 2017.

[90] H. Zhang and V. M. Patel, "Convolutional sparse & low-rank coding-based rain streak removal," *Proceedings - 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017*, pp. 1259–1267, 2017.

[91] E. Ward, Z. Wang, and Q. She, "Generative Adversarial Networks in Computer Vision : A Survey and Taxonomy," no. November, pp. 1–41, 2020.

[92] S. H. Sun, S. P. Fan, and Y. C. F. Wang, "Exploiting image structural similarity for single image rain removal," *2014 IEEE International Conference on Image Processing, ICIP 2014*, pp. 4482–4486, 2014.

[93] P. C. Barnum, S. Narasimhan, and T. Kanade, "Analysis of rain and snow in frequency space," *International Journal of Computer Vision*, vol. 86, no. 2-3, pp. 256–274, 2010.

[94] D. A. Huang, L. W. Kang, Y. C. F. Wang, and C. W. Lin, "Self-learning based image decomposition with applications to single image denoising," *IEEE Transactions on Multimedia*, vol. 16, no. 1, pp. 83–93, 2014.

[95] Y. Luo, Y. Xu, and H. Ji, "Removing rain from a single image via discriminative sparse coding," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2015 Inter, pp. 3397–3405, 2015.

[96] X. Zheng, Y. Liao, W. Guo, X. Fu, and X. Ding, "Single-image-based rain and snow removal using multi-guided filter," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8228 LNCS, no. PART 3, pp. 258–265, 2013.

[97] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan, "Deep Joint Rain Detection and Removal from a Single Image," pp. 1685–1694, 2017.

[98] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley, "Clearing the skies: A deep network architecture for single-image rain removal," *IEEE Transactions on Image Processing*, vol. 26, no. 6, pp. 2944–2956, 2017.

[99] H. Zhang and V. M. Patel, "Density-Aware Single Image De-raining Using a Multi-stream Dense Network," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 695–704, 2018.

[100] B. Li, W. Ren, D. Fu, D. Tao, D. Feng, W. Zeng, and Z. Wang, "Benchmarking Single-Image Dehazing and beyond," *IEEE Transactions on Image Processing*, vol. 28, no. 1, pp. 492–505, 2019.

[101] T. X. Jiang, T. Z. Huang, X. L. Zhao, L. J. Deng, and Y. Wang, "A novel tensor-based video rain streaks removal approach via utilizing discriminatively intrinsic priors," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 2818–2827, 2017.

[102] J. H. Kim, J. Y. Sim, and C. S. Kim, "Video deraining and desnowing using temporal correlation and low-rank matrix completion," *IEEE Transactions on Image Processing*, vol. 24, no. 9, pp. 2658–2670, 2015.

[103] W. Ren, J. Tian, Z. Han, A. Chan, and Y. Tang, "Video desnowing and deraining based on matrix decomposition," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 2838–2847, 2017.

[104] V. Santhaseelan and V. K. Asari, "Utilizing Local Phase Information to Remove Rain from Video," *International Journal of Computer Vision*, vol. 112, no. 1, pp. 71–89, 2015.

[105] A. K. Tripathi and S. Mukhopadhyay, "Removal of rain from videos: a review," *Signal, Image and Video Processing*, vol. 8, no. 8, pp. 1421–1430, 2012.

[106] S. You, R. T. Tan, R. Kawakami, Y. Mukaigawa, and K. Ikeuchi, "Adherent Raindrop Modeling, Detection and Removal in Video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, pp. 1721–1733, 9 2016.

[107] Li-Wei Kang, Chia-Wen Lin, and Yu-Hsiang Fu, "Automatic Single-Image-Based Rain Streaks Removal via Image Decomposition," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1742–1755, 2011.

[108] V. N. Bharath Raj N., "Single Image Haze Removal using a Generative Adversarial Network," *CoRR*, vol. abs/1810.0, 10 2018.

[109] C. O. Ancuti, M.-h. Yang, Y. Chen, B. Chanda, Y.-g. Kim, R. Schettini, J.-p. Tarel, and C. Wang, "NTIRE 2019 Image Dehazing Challenge Report," 2019.

[110] H. Zhang, V. Sindagi, and V. M. Patel, "Image De-raining Using a Conditional Generative Adversarial Network," *CoRR*, vol. abs/1701.0, pp. 1–13, 2017.

[111] T. Alt and J. Weickert, "Learning a generic adaptive wavelet shrinkage function for denoising," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2020-May, no. May 2020, pp. 2018–2022, 2020.

[112] K. Aziz and Y. Himeur, "A Simple and Fast Algorithm for Image Denoising Using the DCT A Simple and Fast Algorithm for Image Denoising Using the DCT," no. October 2020, 2012.

[113] Y. Hel-Or and G. Ben-Artzi, "The role of redundant bases and shrinkage functions in image denoising," *IEEE Transactions on Image Processing*, vol. 30, pp. 3778–3792, 2021.

[114] U. Kamilov, E. Bostan, and M. Unser, "Wavelet shrinkage with consistent cycle spinning generalizes total variation denoising," *IEEE Signal Processing Letters*, vol. 19, no. 4, pp. 187–190, 2012.

[115] A. Azulay and Y. Weiss, "Why do deep convolutional networks generalize so poorly to small image transformations?," 2018.

[116] H. Su, V. Jampani, D. Sun, O. Gallo, E. Learned-Miller, J. Kautz, and U. Amherst, "Pixel-Adaptive Convolutional Neural Networks," tech. rep., 2020.

[117] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," *Proceedings of 2017 International Conference on Engineering and Technology, ICET 2017*, vol. 2018-Janua, pp. 1–6, 2018.

[118] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning book," *Nature*, vol. 29, no. 7553, pp. 1–73, 2019.

[119] Q. Wang, H. Fan, L. Zhu, and Y. Tang, "Deeply supervised face completion with multi-context generative adversarial network," *IEEE Signal Processing Letters*, vol. 26, no. 3, pp. 400–404, 2019.

[120] Cody Marie Wild, "Convolution: an exploration of a familiar operator's deeper roots," 2018.

[121] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," *34th International Conference on Machine Learning, ICML 2017*, vol. 7, pp. 5109–5118, 2017.

[122] N. K. Reblitz-Richardson, V. Miglani, M. Martin, E. Wang, B. Alsallakh, J. Reynolds, A. Melnikov, N. Kliushkina, C. Araya, S. Yan, and Orion, "Captum: A unified and generic model interpretability library for PyTorch," 2020.

[123] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8693 LNCS, no. PART 5, pp. 740–755, 2014.

[124] B. Alsallakh, N. Kokhlikyan, V. Miglani, J. Yuan, and O. Reblitz-Richardson, "Mind the Pad – CNNs can Develop Blind Spots," pp. 1–15, 2020.

[125] S. Li, I. B. Araujo, W. Ren, Z. Wang, E. K. Tokuda, R. H. Junior, R. Cesar-Junior, J. Zhang, X. Guo, and X. Cao, "Single Image Deraining: A Comprehensive Benchmark Analysis," *arXiv preprint arXiv:1903.08558*, 2019.

[126] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 770–778, 2016.

[127] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, (Cambridge, MA, USA), p. 802–810, MIT Press, 2015.

[128] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[129] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a 'completely blind' image quality analyzer," *IEEE Signal Processing Letters*, vol. 20, no. 3, pp. 209–212, 2013.

[130] A. Mittal, A. K. Moorthy, and A. C. Bovik, "No-reference image quality assessment in the spatial domain.," *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 21, no. 12, pp. 4695–708, 2012.

[131] M. A. Saad, A. C. Bovik, and C. Charrier, "Blind image quality assessment: A natural scene statistics approach in the DCT domain," *IEEE Transactions on Image Processing*, vol. 21, no. 8, pp. 3339–3352, 2012.

[132] L. Liu, B. Liu, H. Huang, and A. C. Bovik, "No-reference image quality assessment based on spatial and spectral entropies," *Signal Processing: Image Communication*, vol. 29, no. 8, pp. 856–863, 2014.

[133] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 3296–3305, 2017.

[134] C. H. Bahnsen and T. B. Moeslund, "Rain Removal in Traffic Surveillance : Does it Matter ?," vol. 20, no. 8, pp. 2802–2819, 2020.

[135] X. Zou, F. Xiao, Z. Yu, Y. Li, and Y. J. Lee, "Delving deeper into anti-aliasing in convnets," *International Journal of Computer Vision*, Oct 2022.

[136] Z. Luo, "Alexnet pytorch cifar10." Available at `https://github.com/soapisnotfat/pytorch-cifar10/blob/master/models/AlexNet.py` (2017/12/20).

[137] G. Huang and K. Q. Weinberger, "Densely Connected Convolutional Networks," 2018.

[138] G. Huang, Z. Liu, G. Pleiss, L. Van Der Maaten, and K. Weinberger, "Convolutional Networks with Dense Connectivity," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8828, no. c, pp. 1–1, 2019.

[139] D. Hendrycks, K. Lee, and M. Mazeika, "Using Pre-Training Can Improve Model Robustness and Uncertainty," no. 2018, 2019.

[140] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, Curran Associates, Inc., 2019.

[141] B. Dwyer, "Thermal cheetah computer vision project." Available at `https://universe.roboflow.com/brad-dwyer/thermal-cheetah` (2020/11/12).

[142] M. Kisantal, Z. Wojna, J. Murawski, J. Naruniec, and K. Cho, "Augmentation for small object detection," vol. 2017, pp. 119–133, 2019.

[143] J. Solawetz, "Roboflow coco-2017." Available at `https://blog.roboflow.com/coco-dataset/` (2020/10/18).

[144] G. S. Xia, X. Bai, J. Ding, Z. Zhu, S. Belongie, J. Luo, M. Datcu, M. Pelillo, and L. Zhang, "DOTA: A Large-Scale Dataset for Object Detection in Aerial Images," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3974–3983, 2018.

[145] H. Zhu, H. Wei, B. Li, X. Yuan, and N. Kehtarnavaz, "A review of video object detection: Datasets, metrics and methods," *Applied Sciences (Switzerland)*, vol. 10, no. 21, pp. 1–24, 2020.

[146] D. Du *et al.*, "Visdrone-det2019: The vision meets drone object detection in image challenge results," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 213–226, 2019.

[147] V. Sovrasov, "Sovrasov/flops-counter pytorch: Flops counter for convolutional networks in pytorch framework." Available at `https://github.com/sovrasov/flops-counter.pytorch` (2021/06/12).

[148] I. Sarker, "Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions," *SN Computer Science*, vol. 2, 08 2021.

[149] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, 2019.

CURRICULUM VITAE
Ülkü UZUN

Education

METU, Graduate School of Informatics 2012 – Present
Ph.D. Ankara, Türkiye

Selcuk University, Department of Mathematics 2003 – 2004
M.Sc. Konya, Türkiye

Hacettepe University, Department of Mathematics 1998 – 2003
B.Sc. Ankara, Türkiye

Work Experience

Software Developer December, 2008 – Present
Turkish General Commandership of Gendarmerie Ankara, Türkiye

Awards & Honors

Outstanding Achievement Award
Republic of Türkiye Ministry of National Defence Turkish General Staff 2014

Publications

[1] Ü. Uzun and A. Temizel, "Cycle-Spinning GAN for Raindrop Removal from Images", 16th IEEE Int. Conf. Advanced Video and Signal Based Surveillance (AVSS), 2019, pp. 1-6, doi: 10.1109/AVSS.2019.8909824

[2] Ü. Uzun and A. Temizel, "Cycle-Spinning Convolution for Object Detection", IEEE Access, vol. 10, pp. 76340-76350, 2022, doi: 10.1109/ACCESS.2022.3192022

Specialized Skills

Programming Languages: Python, R, MatLab, Java, C♯

TEZ İZİN FORMU / THESIS PERMISSION FORM

ENSTİTÜ / INSTITUTE

Fen Bilimleri Enstitüsü / Graduate School of Natural and Applied Sciences [ ]

Sosyal Bilimler Enstitüsü / Graduate School of Social Sciences [ ]

Uygulamalı Matematik Enstitüsü / Graduate School of Applied Mathematics [ ]

Enformatik Enstitüsü / Graduate School of Informatics [X]

Deniz Bilimleri Enstitüsü / Graduate School of Marine Sciences [ ]

YAZARIN / AUTHOR

Soyadı / Surname : Uzun
Adı / Name : Ülkü
Bölümü / Department : Bilişim Sistemleri/Information Systems

TEZİN ADI / TITLE OF THE THESIS (**İngilizce** / English) : DERİN ÖGRENMEDE DÖNGÜLÜ ÇEVİRİM YÖNTEMİNİN KULLANIMI/ EMPLOYMENT OF CYCLE-SPINNING IN DEEP LEARNING

TEZİN TÜRÜ / DEGREE: **Yüksek Lisans** / Master [ ] **Doktora** / PhD [X]

1. **Tezin tamamı dünya çapında erişime açılacaktır. /** Release the entire work immediately for access worldwide. [X]
2. **Tez iki yıl süreyle erişime kapalı olacaktır.** / Secure the entire work for patent and/or proprietary purposes for a period of **two year. *** [ ]
3. **Tez altı ay süreyle erişime kapalı olacaktır.** / Secure the entire work for period of **six months**. ***** [ ]

***** *Enstitü Yönetim Kurulu Kararının basılı kopyası tezle birlikte kütüphaneye teslim edilecektir.*
A copy of the Decision of the Institute Administrative Committee will be delivered to the library together with the printed thesis.

Yazarın imzası / Signature **Tarih** / Date 29.11.2022